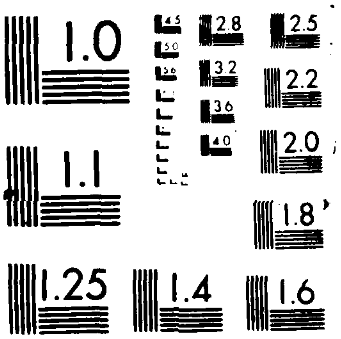
AD-A186 115 USING MULTIPLE DIALOG MODES IN A USER-SYSTEM INTERFACE 1/3
TO ACCOMODATE DIFF..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH R A TAYLOR 1986
UNCLASSIFIED AFIT/CI/NR-87-126D F/G 23/2 NL


1.0
4.5
5.0
5.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/CI/NR 87-126D — 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Using Multiple Dialog Modes in a User-System Interface to Accomodate Different Levels of User Experience: An Experimental Study	5. TYPE OF REPORT & PERIOD COVERED THESIS/DISSERTATION 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Roderick A. Taylor	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS AFIT STUDENT AT: The University of Texas at Austin	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS AFIT/NR WPAFB OH 45433-6583	12. REPORT DATE 1986 13. NUMBER OF PAGES 178
14. MONITORING AGENCY NAME & ADDRESS(If different from Controlling Office)	15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

APPROVED FOR PUBLIC RELEASE: IAW AFR 190-1

Lynn Wolaver 26 Oct 87

LYNN E. WOLAVER
Dean for Research and
Professional Development
AFIT/NR

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

ATTACHED

87 11 10 095

USING MULTIPLE DIALOG MODES IN A USER-SYSTEM INTERFACE TO ACCOMODATE DIFFERENT LEVELS OF USER EXPERIENCE: AN EXPERIMENTAL STUDY

APPROVED BY
SUPERVISORY COMMITTEE:

Michael Wang

Jeffrey A. Brumfield

Eleanor W. Jordan

David Schkade

Joyce J. Elam

Accesion For	
NTIS CRA&I	☑
DTIC TAB	□
Unannounced	□
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and/or Special
A-1	

USING MULTIPLE DIALOG MODES IN A USER-SYSTEM INTERFACE TO ACCOMODATE DIFFERENT LEVELS OF USER EXPERIENCE: AN EXPERIMENTAL STUDY

by

RODERICK A. TAYLOR, B.S., M.C.S.

DISSERTATION
Presented to the Faculty of the Graduate School of
The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN
December 1986

to the United States Air Force,

may it always serve its country proudly

Acknowledgements

I want to thank my good friends and classmates, especially Susan DeBusk, John Isett, Klaus Waibel, and George Widmeyer, who listened to endless discussion about my research and provided valuable support and comments.

I also want to give special recognition to my family for their encouragement and support thoughout this effert. It will be nice to get to know them again.

USING MULTIPLE DIALOG MODES IN A USER-SYSTEM INTERFACE TO ACCOMODATE DIFFERENT LEVELS OF USER EXPERIENCE: AN EXPERIMENTAL STUDY

Publication No.__________

Roderick A. Taylor, Ph.D.
The University of Texas at Austin, 1986

Supervising Professor: Michael Wang

This dissertation investigated a normative theory that says computer users have different dialog needs depending on their level of experience in using a computer. It was developed from the human factors and user-system interface research built upon differences in expert and novice problem soloving strategies, memory, and learning. Experts want to control the interaction with a computer as with a command language whereas novices want to be led through their interaction as with menus and prompts. Since most computer interfaces only provide one dialog mode, some portion of a population having a wide range of user experience is not having their user-system interface needs met. This research hypothesizes that the answer to satisfiy the needs of a mixed population is to have multiple dialog modes that the user is free to choose from and switch between as required.

The hypotheses that experts and novices would perform better and be more satisfied with multiple dialog modes than with just one mode were tested empirically in a controlled laboratory setting. Both novice and expert computer users used one of three types of user-system interfaces (menu, command language, or both modes) to perform the same data base task. Results showed that those with both types of dialog modes were more satisfied and performed better than the command language group but were statistically equal to the menu group even though the performance scores and the satisfaction rating was slightly better for menu. It was found that the subject's choice of dialog mode, when both modes were available, and their satisfaction with a dialog mode had more to do with past experience and preference than with the difference in expert and novice problem solving strategies. Results also suggested that improvements in technology (e.g. fast display times for presenting the menus) and the incorporation of basic underlying functions developed from human factor studies into the user-system interface have tended to overcome some of the previous dissatisfaction by experts for computer directed (menu) type of dialog modes. In conlusion, it was found that even though multiple dialog modes did not improve performance and satifaction over menus (although much superior to having just command language) almost 100% of the 98 subjects expressed a preference for multiple dialog modes. When the

subjects were given a choice of dialog modes the subjects as a whole, regardless of experience, split 60/40, menu to commmand language. Thus, strong consideration should be given to having multiple dialog modes in a user-system interface for application systems that are to be used by a population with a diverse experience background.

Table of Contents

List of Figures

List of Tables

CHAPTER I

Introduction and Problem Definition

All computer systems, no matter what type, require a user-system interface to facilitate communication between the user and the computer system. The interface can range from simple to complex. It can range from setting hardware toggles, to punched cards, to real-time interaction via a terminal. This mandatory interface is necessary for data transfer in or out of the computer, processing instructions, starting/stopping a system or a combination of these plus others. The point is, humans must be provided with a way to communicate with computer systems to accomplish a task and that method is through an interface.

With the advent of interactive systems followed by mini-computers and the micro revolution, emphasis on the user-system interface has grown tremendously. These systems and the current state-of-the-art in hardware has opened the door for a tremendous variety of users who have a wide range of experience, preferences and task requirements. Because of this variety in users' experience and task requirements, many different methods have been used to provide the vehicle for human/computer interaction. Most often, only one communication method (dialog mode) is used in an interface and it is not necessarily chosen because it is the best one for the task or the type of user. The determination of which dialog mode to use is many times based on nothing more than what is easiest for the development team to implement. Other times it is based on what the designer feels the typical user would prefer as a dialog mode or on what the designer has seen in other interfaces. In any case, the reasons for selecting one mode over another are quite varied but they all have the basic underlying problem that all the users, regardless of their experience with computers or the problem task, will have to learn/conform to the selected dialog mode if they want to use the system.

This current insistence on providing one dialog mode for all classes of users is a common practice that needs to be critically examined. Research in cognitive psychology, information systems and computer science has provided indications that the dialog modes best suited for a user-system interface used by novices do not necessarily meet the needs and requirements of experienced users. This seems to be based on such cognitive issues as short and long term memory, learning, human information processing, and the differences in expert and novice problem solving strategies. Because of this, designers must become more cognizant of the user's needs in these areas and reflect these needs and differences in the user-system interface. This can be done by: (1) designing interfaces that contain a fundamental set of underlying functions that meet the needs of all users and (2) providing flexibility in the dialog mode to accommodate users with different levels of experience. Without these two things, a user may have to use a system whose interface is confusing or stilted causing the user to misunderstand or misuse the system in accomplishing their task.

This need for a fundamental set of underlying concepts is well supported in the literature both theoretically and empirically. The need to accommodate different levels of user experience is consistently written about in a normative manner but there is little empirical evidence to accompany it. Because of this, the main thrust of this research is to provide empirical evidence for the need to accommodate different levels of user experience in a user-system interface for application systems. The research is designed to determine whether or not the use of multiple dialog modes can accommodate the differences in user experience and result in higher user performance and satisfaction than an interface with only one dialog mode.

Interface Definition and Components

To properly discuss the implications of the user's performance (effectiveness and efficiency in accomplishing a task) and satisfaction (perceived usefulness of an interface to allow the accomplishment of a task) with a user-system interface, one must consider different aspects of human information processing as well as the interface itself. It is not sufficient to just consider what "looks good" in an interface without also trying to maximize a user's performance and satisfaction. Therefore, it is important to know what the different components of an interface are so that each piece can be analyzed as to its effects on user performance and satisfaction and provide a foundation for further discussion of user-system interfaces.

As discussed earlier, an interface is the communication medium through which a human and a computer carry on a dialog to accomplish a task. Based on the work of H. Smith [80] and Benbasat et al [81], the interface can be conceptually viewed as having four components (Figure 1.1).

Smith breaks the communication between humans and computers (the interface) into three components: tasks, users, and communication interfaces. Benbasat et al list four categories: human user, decision environment, task, and interface characteristics. Using primarily Smith's definition of an interface as a baseline, this research generalizes his first two components into dialog task and user experience while the third component is split into dialog mode and underlying concepts. It is these four basic components that provide the basis for discussing user-system interfaces.

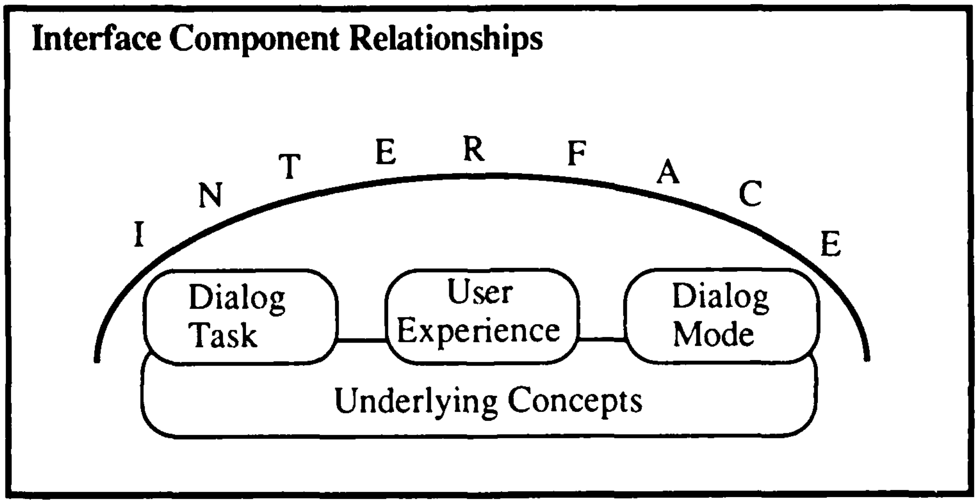


Figure 1.1

Dialog Task: Smith's original use of task related more to the domain of a specific problem or application and the functionality of the application. Later, Benbasat et al generalized the term somewhat so it applied to a category of applications such as CAI, decision support and word processing instead of one specific application. In this research, this term will be generalized even more to reflect the more generic lower level task operations a computer user would perform or need to comı ınicate while working within an application or problem domain. These generic tasks are such things as data entry, text editing, query retrieval and process control. This is not an exhaustive list but represents the major types of activities that a computer user would perform when interacting with an application to accomplish some task.

User Experience: Smith's definition of a user was based on the pre-'80s concept of how computers were used. He defined three groups: those who built systems (system support users), those who used the applications (end users) and a large group between these two (mid users). While this categorization was appropriate then, the micro computer explosion has caused one to rethink who a user is and how one should categorize him. It is no longer appropriate to group users by how they use the computer but more on their experience with using computers and working within the problem domain.

The categories in which one must consider the experience of a user are: (1) technology, (2) problem domain (3) application specific and (4) general. Within each of these categories a user's experience can range from naive to causal to expert or anywhere in-between.

(1). Technology Experience: This area applies to a user's experience with computers regardless of the computer's size or type, the problem domain or the application domain. The level of experience in this category can be quantified by a combination of frequency of using computers, number of years the person has been using computers and individual's computer education and knowledge.

(2). Problem Domain Experience: This applies to a user's experience in the problem domain without regard to computers. For example, an individual can be an experienced accountant whether he keeps the books by hand or uses a computer product. Frequency and number of years working in the problem domain as well as education and knowledge can be used to categorize a user's experience level.

(3). Application Specific Experience: The experience level here is closely linked to both the problem domain and technology. It reflects the user's experience in using a particular application that pertains to the problem domain. It can be measured in a similar fashion to technology and problem domain.

(4). General Experience: This is a user's general knowledge that has been acquired through life's experiences, education, society's norms, etc. It does not have a range of experience per se that can be easily measured as the other two categories but it can impact one's attitudes and perceptions in gaining experience in the technology and problem categories.

Figure 1.2 shows several possible relationships between the different experience categories for a particular problem domain and a user. Other users may have the same or different relationships. In the figure, the larger the diameter of a circle the more experience the user has in the category relative to the other categories. The first example, (a), represents a high level of experience in all areas. (b) indicates that a user has considerable problem domain experience but little technology

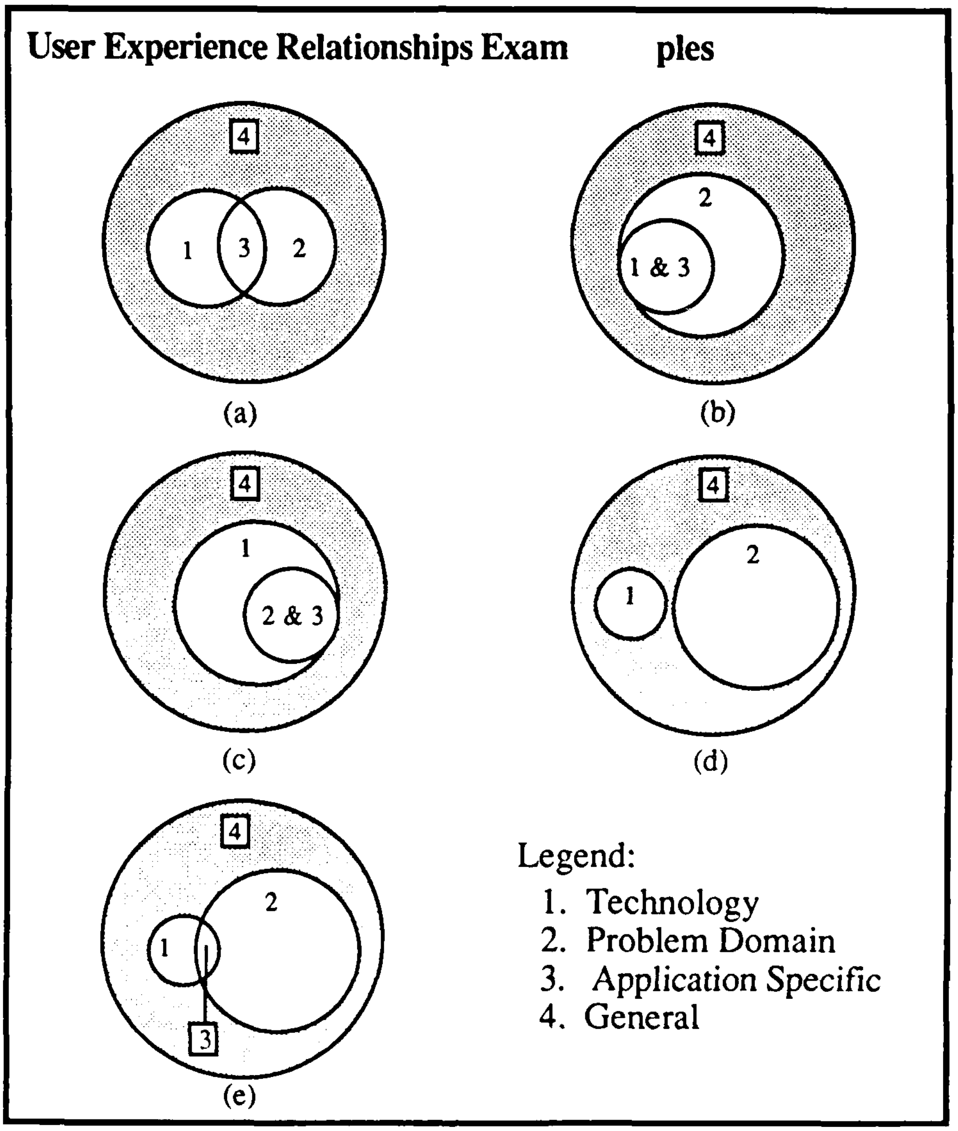


Figure 1.2

experience all of which is experience with a particular application in the problem domain. (c) is the opposite of (b) where the only experience in the problem domain is with an application. (d) and (e) represent the typical experience levels for technology and application novices.

When considering computer interfaces, the user population may contain any combination of "experiences" from these categories. Users could easily be experts in the problem but novices in technology or vice versa. These combinations of different experience levels can have a definite impact on the interface needs of the user population (Larson [82], Eason [76], Maguire [82]).

Dialog Mode: This is the vehicle for representing the communication between the user and the computer. The representation used must be able to convey the meaning and the intent of both the computer's and the user's requests and actions. Studies have shown that while any dialog mode can eventually be mastered, the user's technology and application specific level of experience can have a bearing on which mode is most suitable and most preferred (Smith [80], Shneiderman [80], Savage [84]).

There are several different dialog modes that can be used in an interface. The two most often used, and the two that will be studied in this research, are menus and command languages. Some of the other dialog modes are natural language, question and answer, and forms.

(1). Menu: This is probably the most popular form of communication with an application. It consists of lists of options presented to the user from which he makes a selection. The selected item may perform an action or generate another menu for eliciting additional information. Menus have the advantage of restricting the domain presented to the user which aids in error control. By presenting all of the information in lists, users are able to learn the application more readily and be able to be productive with little to no training (Hall [82], Shneiderman [79]). There can also be a high degree of control by the application since the user's dialog is restricted to the menu presented by the application. Interfaces that use this type of dialog mode control the flow of communications and are called **computer directed dialogs**.

(2). Command Language: This dialog mode generally consists of verb - noun pairs followed by options. This mode puts the user in control of the flow of dialog and is an example of **user directed dialog**. While this mode is generally harder to learn and can require extensive training, it provides the user the freedom to control the interaction and not be restricted to a particular subdomain (as long as they are legal or valid). This mode relieves the user of having to see lists of choices when he already knows the choices and knows what he wants to do. It is a more succinct form of communication.

Underlying Concepts: This is the foundation on which an interface is based. It consists of functions that implement procedural and conceptual requirements that have been developed through experience and human factors research. While the human factors research area is very broad and varied, there are many procedural and conceptual concepts in this area that have been shown to be applicable to all user-system interfaces regardless of the type of dialog mode used, the dialog task performed or the user's experience. This subset of the literature will be collectively called the interface's underlying concepts for this research (see Figure 1.3 for examples). It is for the above reasons that Smith's "communication interfaces" and Benbasat et al's "interface characteristics" are spit into dialog mode and underlying concepts. While these underlying concepts generally transcend dialog mode, dialog task, and user experience, these three interface components none-the-less have an impact on the degree to which the underlying concept is implemented in an interface. For example, is has been shown that an interface requires some sort of help function (to assist in memory and learning) but the quantity of information displayed and the method

Underlying Concepts Examples

Conceptual	Procedural
reliability	standard abbreviations
consistency	error messages
simplicity	object naming
feedback	help
predictability	current state
adaptability	progress indicators
personalization	spelling correction
tolerance	menu formats
mental load	command language syntax
type of user	dialog control
ease of use	cancel/undo
self-sufficiency	synonyms
⋮	⋮

Figure 1.3

for obtaining the help information is dependent upon user experience (Magers [83], Black [81], Paxton [84]).

Since this part of an interface can influence or be influenced by the other three components, its discussion is somewhat longer. It is also the component that has received the most attention from researchers with the result being that many of the syntactic problems associated with this area have been solved (though not necessarily implemented into interface design). For these reasons this research assumes that this foundation of underlying concepts exists and concentrates more on the other interface components. The following discussion is presented because it is important to understanding all the components of an interface and to provide the reader with a basis for later discussion.

Much of the earlier literature on this subject is based on experience, intuition, common sense and to some extent personal opinion. But as this research area has matured, experiments and concepts grounded in theory have been developed. First of all, there have been a number of "concepts" lists describing the necessary ingredients and procedures for having a successful interface. Most often they are presented in conjunction with interface design discussions. Shneiderman [80] identifies eight different lists of interactive interface design considerations and goals that were developed in the '70's. They contain both procedural and conceptual concepts that were developed primarily from experience and opinion although many of them have since been verified through further research. The key elements that flowed through all of these lists are: simplicity, good error diagnostics, good help functions, consistency, and knowledge of the type of users. Shneiderman also presents several broad human factors issues that are important to consider in developing any type of interactive system regardless of the dialog mode or dialog task.

(1). Attitude and anxiety - Studies have shown that these can dramatically reduce learning and interfere with performance. An interface must be designed to alleviate these problems.

(2). Control - A driving force in human behavior is the desire to control. With computers, the desire to control tends to increase with experience. Novices have been found to willingly accept the computer as the controlling agent (computer directed dialog). Evidence to support this issue of control was experimentally obtained by a longitudinal study of novices performing an editing task over a thirty day period (Gilfoil [84]). After about 18 hours of experience on the task, the subjects had all stopped using the computer directed dialog (menus) and had switched to a user directed dialog (command language) for the remainder of the study.

(3). Closure - the completion of a task leading to relief and the release of short term memory (discussed in next section) for some other task. This has the implication that novices may want multiple small tasks rather than one large task and has been borne out in a recent study (Kiger [84]).

(4). Assistance - [error handling in the reference] multiple levels of HELP depending on the experience of the user. The studies on different levels of help systems show this to be the an important issue.

More recently, Hayes, Ball and Reddy [81] listed the following items as necessary capabilities for an interface. These items are along the same lines as Shneiderman's:

(1). Flexible parsing - tolerant of typing errors.

(2). Robust communication - must be able to talk to different classes of users.

(3). Explanation system - both static and dynamic information.

(4). Personalization - conform interface to user(s).

Gains and Shaw [83] made a detailed review of dialog programming rules and developed a number of rules (actually conceptual and procedural concepts that need to be included in an interface) that are categorized into four broad areas: (1) minimizing the mental load on the user, (2) error detection and correction, (3) systems analysis and development, and (4) user adaptation to the system. The first two are particularly applicable to this research and can find their basis primarily in the human memory, learning and information processing processes.

(1). Minimizing the mental load on the user:

- uniformity and consistency in terms and operations.
- provide memory aids at all levels.
- train through experience.
- make the state of the dialog observable.

(2). Error detection and correction

- validate data on entry.
- provide a re-set command.
- provide a backtrack facility.

One of the more comprehensive set of guidelines for developing computer interfaces has been developed by Foley and Sibert [83]. They present a wide range of design principles, methods and guidelines that touch on all aspects of developing an interface. Of particular interest are the underlying concepts (called guidelines in reference) that apply to any interface. These concepts are both procedural and conceptual in nature:

(1). Provide feedback

(2). Help user learn system

(3). Provide error correction and prevention

(4). Control response time

(5). Design for consistency

(6). Avoid memorization

(7). Avoid unnecessary spatial-linguistic conversions

James [81] provides four broad conceptual guidelines that seem to capture the essence of many of the other more specific concepts. The first two, reliability and adaptability, are the same as others have defined but with the addition that adaptability includes providing an interface that is consistent with the user's previous experiences. The third concept, self-sufficiency, means that the user does not have to rely on any other source to perform his task. His final concept is ease of use. On this he says:

> ...ease of use is dependent on the individual background of each user. There is no particular style of use which can be correct for all users. This implies that the system must cater for a wide range of styles. In particular it must cater for and support the inexperienced user with information on how to

> proceed. At the same time, it must permit the experienced user to develop a personal, idiosyncratic style which represents optimum performance.

What James is saying, along with many other researchers is that while there are these underlying concepts that apply to all types of users, the interface must also provide different dialog modes to accommodate the differences between inexperienced and experienced users. And, it is because of these user differences that there is a different degree to which some of these underlying concepts should be implemented.

Research Questions

Based on the above discussion, prior research and theoretical foundations (both to be discussed in the next chapter), there are two questions this research will concentrate on. The first question is based on the need to accommodate different levels of user experience. The second concerns itself with the amount of experience a user needs to acquire before wanting an interface that is not oriented toward a novices's requirements.

1. Will the availability of multiple dialog modes in a user-system interface that accommodates different levels of user experience make a difference in overall user performance and satisfaction?

2. At what level of experience will a user prefer a user directed dialog over a computer directed dialog?

Investigative Questions

Although there are a number of specific questions that could be asked to answer each of the above research questions, this research will limit itself to the following as they apply to user-system interfaces for applications systems:

1. Will an interface that provides multiple dialog modes for different user experience levels improve a user populations' perform ınce and satisfaction over an interface with a single dialog mode?

2. Does user experience level make a difference in the user's task performance and satisfaction for different types of dialog modes?

3. At what level of application experience will a user prefer an application command language over an application specific menu dialog mode?

4. What is the relationship of technology experience to application specific experience in determining whether or not a user is more satisfied with a computer directed interface versus a user directed interface?

Summary

This introduction has shown that dialog mode, dialog task, user experience and underlying concepts all play an important role in the usability of the interface to achieve the accomplishment of a task. It has been pointed out that while these four elements make up the components of an interface, differences in expert and novice problem solving and the cognitive issues of memory and learning must be considered in concert with these four elements. An important question that arises from these interactions is how do all these elements interact with each other and which ones have the most effect on achieving the goal of the user-system interface to facilitate communication that will maximizing user performance and satisfaction in accomplishing a task. It is also important to know which component can be manipulated (without ignoring the cognitive processes) to best achieve the interface goals. The next chapter will consider these questions and provide a model for analyzing these relationships after discussing the relevant literature and theoretical foundation for the research. The final section of the following chapter will discuss the results of prior experiments that relate to user-system interfaces and varied experience levels of user population.

CHAPTER II

Prior Research and Theoretical Foundations

One of the main themes that flows through most of the literature on user-system interfaces is a notion that user experience does make a difference in the interface requirements and should be taken into consideration. This notion, based on an accumulation of evidence in the form of opinion and some experimentation on the differences in a user's interface needs depending on their experience level, suggests a normative theory for maximizing performance and satisfaction. Support for this normative theory can be found in the literature and is discussed in the next section. In the second section, support for the normative theory is developed from the theoretical basis of the differences in expert and novice problem solving strategies, memory, and learning. The third section analyzes the user-system interface in light of the need to accommodate different levels of user experience and suggests that the key to maximizing performance and satisfaction in a user-system interface lies with the interaction between the user experience level and the dialog mode with the caveat that the user-system interface contains the necessary fundamental underlying concepts. The final section discusses some of the previous experiments that provide evidence for the normative theory. The research hypotheses developed from the literature, theory and past experiments are discussed in the next chapter along with the experimental design.

Literature Review

Much of what is in the literature on the need for an interface to support both novice and expert users are opinions by a number of researchers from cognitive psychology, information systems, and computer science (Norman [81], Chapanis [82], Black and Sebrecht [81], Stevens [83], Hall [82], Benbasat and Wand [84], Larson [82], and Maguire [82]). These researchers openly advocate this need and have said such things as stated by Norman as one of his major conclusions concerning human-computer interaction in his book on memory and learning:

> A designer [of interactive systems] must think of both the beginner and the expert simultaneously. The expert wants a simple system with a minimum of intrusion in terms of help and prompting and extra responses. The beginner needs continual prompting, reassurance, extra responses that confirm or allow for a change of mind. To design one system for all classes of users is not easy, but it must be done.

In a later paper, Norman [83] takes this line of thought even further when doing a tradeoff analysis between menu and command language dialog modes. He concluded that the menu (computer directed) dialog is best for novices and that command language (user-directed) dialog is best for experts thus implying the need for multiple dialog modes to support the different user experience levels. Chapanis [82; pg 113] said the following while trying to answer a question about who the real user of a computer system is and what his experience is: "The answer, more often that not, is that there is no single user, but that every system has multiple users and that all of them need to be considered...."

He then goes on to discuss how, using different dialog modes and underlying concepts based on human factors, an interface can consider different types of users. Likewise, in Black and Sebrecht's [81; pg 164-165] discussion on how basic cognitive psychology can provide guidelines to design interactive systems that are easy to use, they state in their section on "User [experience] Level" that there is a need for "flexibility" in the interface because:

> Insofar as there is no single well-defined user type, it is important to design systems with sufficient flexibility to accommodate different users....There are two types of flexibility that need to be considered: between group flexibility and within group flexibility. The first concerns the extent to which the system is capable of differential response to two [different] groups of users.... A...solution to these differences in user ability is to provide two or three different levels of use [dialog modes and underlying concepts] and to allow the user to select the appropriate level.

Beside just supporting the need to accommodate different levels of user experience, there is also support for the interface goals of maximizing performance and satisfaction. In a critical evaluation of interfaces, Stevens [83; pg 8] says: "A software interface designed on a naive-user basis will frequently be an irritating encumbrance to one familiar with the system and a consequent information overload effect may even prevent the full capabilities of the system being utilized." He goes on to say: "It is not user-friendly to make a system easy now if it is later limiting in terms of reasonable expectations of a user's increase in proficiency over time." Supporting this same line of thought, Hall [82:

pg 447] makes the follow statement when describing an interface design methodology to accommodate multiple level of user experience:

> ...there is a need to accommodate both experienced users, who require a very succinct language to enable them to work fast and without frustration [user directed], and naive users who need a lot of assistance [computer directed]. It is desirable for the naive user to progress to advanced status without having to learn a new advanced dialogue separate from their beginner's dialog. HELP commands, abbreviations for Er glish-like keywords, and MACROS...are widely used: these ideas are useful, and they do accommodate a range of competences, but can we not go further?...it should be possible to enable the user to freely switch from menus or forms to programming language [command language] within a single system.

In Benbasat and Wand's [84; pg 106] paper on designing human-computer dialogs, they draw the same conclusion as Hall. They say that users have different needs in both human-computer dialog and the degree in which functions (underlying concepts) are implemented:

> While novice users would prefer a computer-guided [computer directed] system with extensive help and error correction facilities, the experienced users would like a user-guided [user directed] system which accepts short messages and allows the entry of multiple commands to speed up the dialogue. In other words, the quality and effectiveness of the dialogue type chosen depends on the person (Thimbley,1980). The designer has also to consider that over time...the novices would become more experienced users. There is, therefore, a need for a dialog generator [i.e. a design methodology that would generate different dialog modes] which could be used to change the

> human-computer interface efficiently to suit the evolving needs of the users and the demands of users who have different levels of expertise.

In a similar fashion Larson [82; pg 439] says:

> Menus ... can be a waste of time for the expert user, who is already aware of the options on each menu. Experienced use.:s prefer to enter commands directly by typing keywords and phrases onto a single line on a terminal - a very difficult task for novices....A method is needed that gives the novice his menus, yet allows the experienced user to keyboard his input directly [computer directed verses user directed dialogs]. Such a method would expand rather than restrict the user's freedom in formulating commands....

From a survey of published recommendations on designing interfaces, Maguire [82] found that the consensus of the literature states that the user-system interface should contain two levels of dialog, one for experienced the other for inexperienced.

There are many instances of other researchers discussing this issue that an interface should provide support for different levels of user experience (Benbasat et al [81], Bertino [85], Eason [76], Gains [81], Hayes et al [81], Hiltz [84], James [78], Kuo [85], Nievergelt [82], J. Martin [73], Mozeico [82], Paxton [84], Robertson [85], Savage [84], Shneiderman [79], H. Smith [80], Treu [77], Wasserman [81]). They all concern themselves with the problem of novice users having different user-system interface needs than do experts and that these needs are many times opposites. Without accommodating the different needs between novices and experts (or experienced versus inexperienced), one or both groups will be faced with having to use interface functions or modes that are inappropriate for their level of experience. The inappropriateness could inhibit the user from maximizing both his performance on the problem task and his satisfaction with the interface.

This type of support for accommodating different levels of user experience is pervasive and can be stated in terms of a normative theory even though none of these researchers have yet substantiated or validated it. Thi; theory, based both directly and indirectly on the differences in information processing found at different levels of user experience (though not empirically validated), can be stated as follows:

> An interface should provide support for the needs of users at different experience levels if it is to maximize performance and satisfaction while providing the best facilities possible for assisting the user in accomplishing his task. This support can best be provided through dialog modes and underlying concepts that can be manipulated to satisfy the different user requirements.

Theoretical Foundations

The following discussion provides a theoretical basis for the normative theory developed from the literature review. The first area, memory and learning is an integral process of human information processing that leads one from an inexperience state to some higher experience level. Using this theory as a basis, one can investigate the differences in expert and novice problem solving strategies and relate it to user needs in the user-system interface.

As with any activity someone performs, using a computer interface also involves the human process of memory and learning. James [81] states: "Providing a good user interface is basically concerned with human patterns of learning, using, remembering and forgetting." They are important to consider because they are the processes that can lead a user from naivete to an an eventual expert. They are continually being used both consciously and unconsciously in all activities including interacting with a computer interface. Because memory and learning are so closely related, a brief description of each is presented before relating them directly to user-system interfaces and users.

Memory: Memory consists of two basic functions: information storage and processes for manipulating the stored information. One way to describe these two memory features is through an Information Processing Model (Mayer [81]) as shown in Figure 2.1. The functions in the model are :

(1). Short-Term Memory (STM) and Working Memory (WM): These two memories together (hereafter referred to as STM collectively) are basically our conscious memory. It holds all the information that a person can be aware of at any one time. Experimental evidence has shown STM to be of limited capacity (Norman [81] and Miller [56]). This capacity is usually described as about seven separate 'chunks' of information where the amount of data in a chunk can vary considerably (Miller [56]). New information received in STM will replace existing information causing it to be lost unless other processes are used to retain

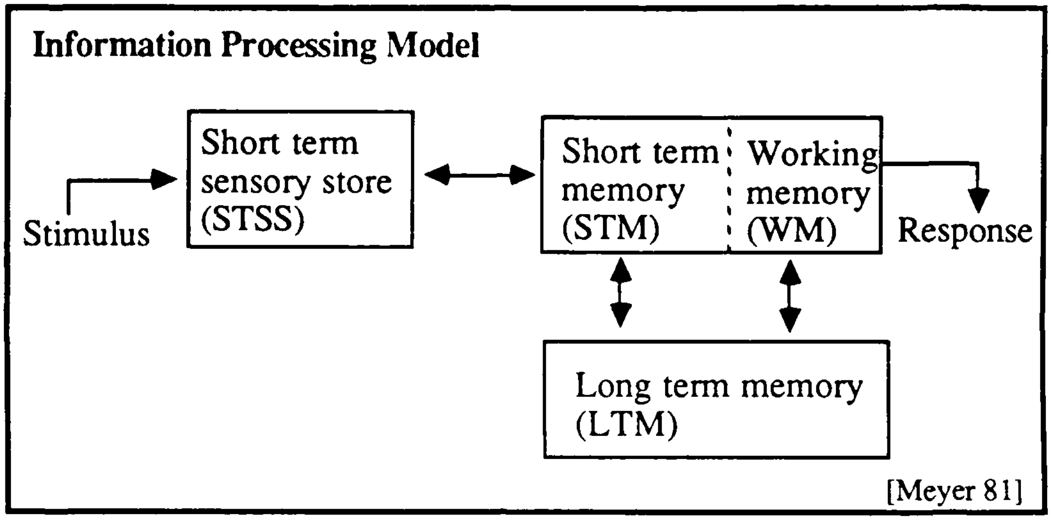


Figure 2.1

the information such as rehearsal or encoding. The rehearsal process can keep information in STM and the encoding process can transfer information from STM to long-term memory.

(2). Long-Term Memory (LTM): The capacity of this store is considered unlimited. The organization of LTM is still a matter of speculation but it can be said that it is organized in some pattern and items in LTM can be retrieved into STM by following some search path. LTM does not fade with time but items can be blocked for retrieval when new information interferes with retrieval paths (Norman [81], Mayer [81]).

(3). Short-Term Sensory Store (STSS): This is the storage place for raw information received by our senses. While seeming to be unlimited in capacity, information in this store fades quickly unless transferred into STM (Mayer [81], Card [83]).

Learning: Hand-in-hand with memory is learning. Learning is the process of organizing LTM such that the acquisition and retrieval of information can be successfully accomplished. Learning can be thought of as having three processes: (1) accretion, (2) structuring and (3) tuning (Norman [81]).

(1). Accretion: This is the addition of new knowledge to existing memory structures. It is the process of gradual accumulation of knowledge. Ideally, the new knowledge fits within a prior framework of knowledge that is appropriate for organizing and maintaining the information. At other times the fit is bad and the new knowledge is apt to be tucked away into inaccessible areas of memory or interpreted in an inappropriate manner until some additional knowledge corrects the situation. An example of accretion is when a user who is experienced with word processing tries a new text editor. This user would already have a structure for text editing and would only be learning new names (commands) to perform familiar function.

(2). Structuring: This is the formation of new conceptual structures and conceptualizations. Restructuring of knowledge is an infrequent occurrence but when it occurs it leads to fundamental improvements in understanding. Structuring of knowledge affects understanding and the retrieval process. Structuring can be part of that step that occurs when someone's actions for accomplishing a task becomes more of a compiled process that occurs automatically rather than being a number of individual steps (e.g. part of the process for going from novice to expert).

(3). Tuning: These are fine adjustments made to stored knowledge for accomplishing a task. Practice is the process for accomplishing tuning and is the learning activity that eventually leads to expertise.

These three learning processes, accretion, structuring, and tuning, are used to describe learning in any setting and can therefore apply to learning to use a user-system interface. While these are the fundamental processes in learning, a higher viewpoint of learning as it applies to interfaces was developed by Treu [77]. He described four learning stages a user progresses through when learning to use an interactive system (his example system was a graphical system). His primary motivation for describing these learning stages was to show why the interface dialog mode and underlying concepts degree of functionality need to be adjusted to correspond to the user's ability at each of the stages. These steps, applicable to user-system interfaces in other problem domains as well, describe the progression that a user progresses through as he moves from a novice to an experience user:

(1). Learning the basics of the system.

(2). Progressing to more independent use of the system.

(3). Probing into the more subtle or difficult features.

(4). Producing quality results within known system constraints.

Mozeico [82] built upon Treu's four stages and added a fifth stage to describe infrequent and casual users who do not progress though all of the stages. He felt that Treu only reflected the stages of a motivated user striving for mastery of the system. He places this fifth stage at about the same place as stage 1:

(1a). Using the system to obtain desired results with minimal prerequisite knowledge.

A system built by Mozeico to test his theory used a question and answer and a tutorial dialog mode for stages 1 and 1a. A command language mode was used for the other stages. Initial testing of the system showed some success except that the command language for stages 2, 3 and 4 proved to be too difficult a transition from stage 1. He felt that a menu dialog for stage 2 would provide the better transition to a command language dialog in the last two stages.

While Treu and Mozeico have applied learning stages to the requirement for different dialog modes to accommodate the user's needs at each of the stages, other researchers have used a single dialog mode to determine the effects of learning on the user's ability to accomplish different types of activities. An experiment with novice users in computer technology showed that they preferred and performed better with a menu system that contained only 5 to 8 items per menu than either fewer or more items per menu (seven chunks) (Kiger [84]). Another experiment showed that one menu of 8 items per screen was preferred to three menus per screen, 8 items each, even though the three menus per screen reduced the overall processing time (Kiger [84], Savage [84]). For novices, this suggests that each item on the menu filled a chunk and that any more surpassed the capacity of STM. It also showed that the novices were using subgoals to accomplish the task; each menu was a subgoal. When a similar experiment was done with experts, it was found that each menu was considered as a whole chunk and not just each item on a menu list (Savage [84]). The experts seemed to have restructured the information into much larger chunks than the novices.

From this brief overview of memory and learning, it is easier to see how a correctly designed menu system can be beneficial to novice users. Each menu provides limited information that can fit into STM. The menu list itself provides information to assist in the retrieval of information from LTM so the user can make a decision on selecting the correct menu item for accomplishing his task. For experienced users however, menus can be a source of frustration (Bertino [85], Stevens [83], Norman [83]). Experienced users (experts) seem to have structured the information such that each menu is a chunk and therefore do not need or necessarily want to receive the menu list. They already know their course of action and do not need the menu list to help with this decision. They just need to be able to express their need.

The opposite type of results have been found to be true with command languages. Novices tend to do much poorer in accomplishing a task when using a command language dialog mode then do experts. Evidence has shown that the experts tend to group the command by activity/function and novices do not; they recall commands individually (Gilfoil [84]) The clustering of commands by function allows experts to recall groups of command while novices have each command as a separate entity. This suggests that experts can retrieve a process, consisting of one or more commands, into a chunk in STM freeing up the rest of STM for other actions. Novices on the other hand, only retrieve a command into each chunk of STM. For novices, running through a mental list of seemingly unrelated commands to find the correct one(s) for performing a certain task can be difficult and frustrating.

Expert Versus Novice: From the discussion on user experience levels in the previous chapter, one can see that individuals can range from novice to expert in both the problem domain and in computer technology. Much has been written about the differences between novices and experts in their problem solving processes and abilities in different problem domains such as chess (Simon [84]), physics (Larkin, McDermott, Simon and Simon [80]) and text editing (Card, Moran and Newell [83]). These studies have pointed out a number of differences between novices and experts. Foremost in differences is the amount of knowledge that an expert has gained through studying and experience. Hayes' [86] study of great painters and composers who are considered to be experts in their fields revealed that it took about 10 years to reach the lofty status of expert. Others studies have shown that experts seem to be able to directly recognize situations and apply a solution. Many of their actions are automatic and seem to be more of a compiled process (Simon [84]) that is invoked almost unconsciously. The same cannot be said of novices

An interesting point about expert's performance is that while it is consistently better then a novice's within the experts problem domain (Larkin et al [80]), it is no better than novices when put into a situation outside the expert's domain. The effect is that individuals can be an expert in one domain and a novice in another. The needs and problem solving abilities of individuals can change depending on the domain. For user-system interfaces, this means that a user can be an expert for one application but a novice for another regardless of their technology experience and thus his interface needs may be different for each of the application domains.

Novices are usually characterized as being the opposite of experts in all of the above areas. They have limited training, experience or knowledge or a combination of all three in the problem domain. One distinct characteristic that distinguishes novice problem solving from that of an expert's is that novices seem to be subgoal oriented in their problem solving. They start with the basic facts and progress in a forward chaining fashion toward the final solution by solving subgoals. They do not have a precompiled process to apply to the solution. This idea of solving subgoals relates back to Shneiderman's "closure" concept. Experts on the other hand tend to look at the final goal and apply some process to accomplish that goal (Larkin et al [84]) or use backward chaining to reach a point where an existing processes can be applied.

These differences between novices and experts have also been evaluated in problem domains where the computer user-system interface is an integral part of generating a solution to the problem. In Card et al [83] in-depth experimental studies on experts and novices performing text editing with a word processor, they found that the novices would rather use several basic steps to perform an editing task (use of subgoals) to the expert's desire to use one specific command (a compiled process) to perform the same editing task.

What the literature shows is that experts and novices use different methods to complete tasks. They have different amounts of knowledge and experience to apply to accomplishing the task and approach the task differently; i.e. subgoal versus compiled process that is goal oriented. Applying this to a user-system interface, and dialog modes specifically, it seems that novices would be more inclined to prefer and actually perform better if they used a dialog mode consistent with their problem solvin;; strategy such as menus. Menus are a form of subgoal processing since each menu is in itself a subgoal that leads the user to the final solution (goal). This same reasoning applies to experts as well except they could use a command language to accomplish the goal directly. With menus, the experts can become frustrated with the method (a subgoal process oriented to novice style of problem solving) for accomplishing the task seemingly because that is not their problem solving process (Stevens [83], Larson [82]).

The importance of the difference between expert and novice needs cannot be over stated. The impact on user performance and satisfaction when these differences are either ignored or satisfied is an important issue that concerns this research.

Conceptual Interface: Given that the normative theory derived from the literature exists and is correct, what is available in an interface that would allow the designer to implement the theory? To answer this question, each component of a user-system interface needs to be analyzed for the possibility of being manipulated in support of the theory without affecting the user-system goal of maximizing performance and satisfaction. The components must be evaluated in relationship to the other components and with the overall effects that can occur when it is manipulated. In Chapter I, a conceptual diagram (Figure 1.1) was shown to describe the basic relationship of the four key components in an interface: dialog mode, dialog task and user experience all built upon a foundation of underlying concepts. While this model showed relationships, it did not consider the interaction of these elements with each other and their effect on the user in accomplishing his task. (Remember, an interface's only purpose is to facilitate a dialog between a user and a computer for the express purpose of accomplishing or solving some task). One way to view the interaction of these elements is through a three dimensional model (Figure 2.2). The user experience dimension is a continuum from novice to expert on all four of the categories of user experience. The dialog mode and dialog task dimensions are a set of discrete values with no continuum implied by the examples used in the figure.

If a location in the three dimensions is selected, then it is the combined interaction of the three dimensions along with the specific and general underlying concepts available at that point (the fourth dimension) that determines a user's performance and satisfaction for accomplishing his task. The ideal situation then would be to find that point in the three dimensions that would determine the interface characteristics for maximizing the user-system goal for the given problem task and given user. Finding the ideal point assumes that each dimension can be constrained to a single value which may not be feasible and that the four

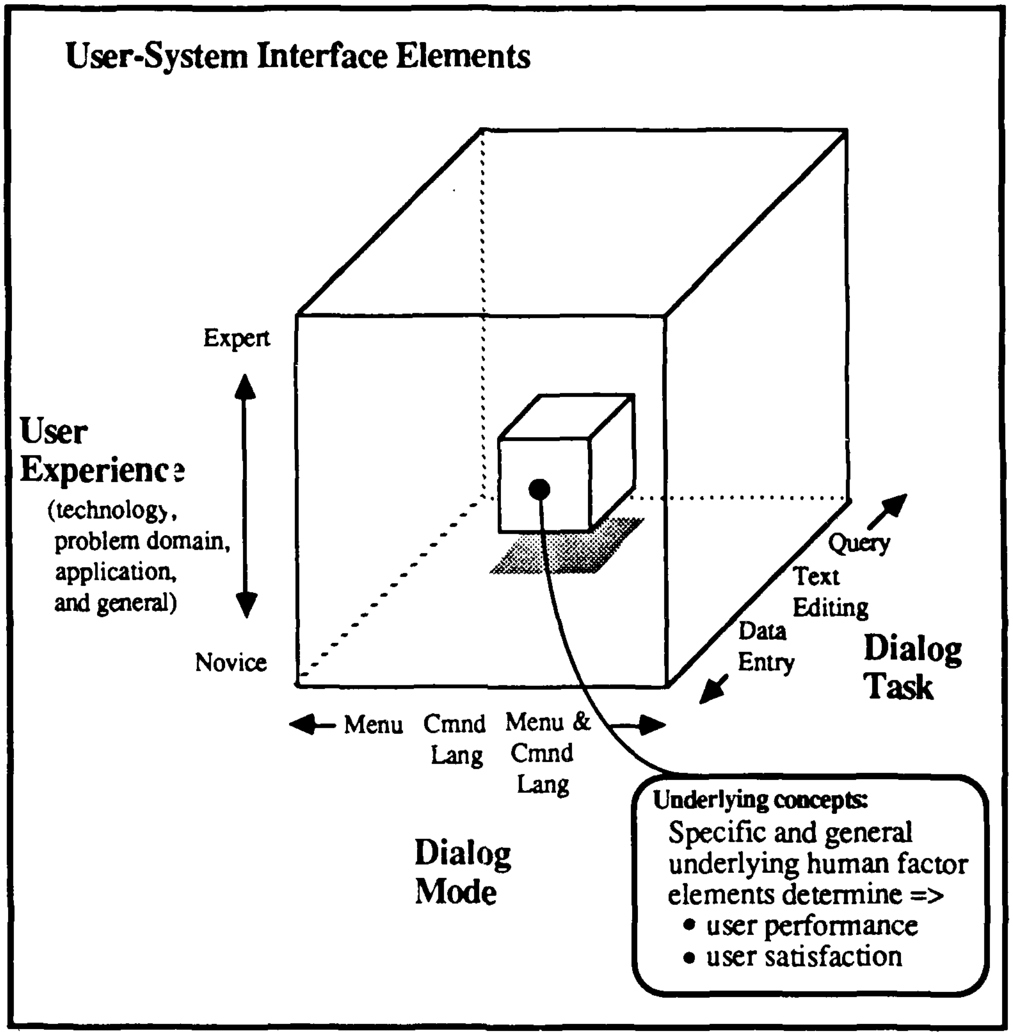


Figure 2.2

elements are independent both of each other and the problem such that they could be manipulated freely to maximize the interface goals which may not be the case.

In the following section, each of the dimensions will be discussed as to their degree of "flexibility" to be manipulated in designing and developing an interface such that user performance and satisfaction are maximized. It is only through manipulations of these dimensions that one can attempt to achieve maximum user performance and satisfaction in accomplishing the problem task while still accommodating different levels of user experience. From this discussion, it will be shown that the best and most viable dimension to manipulate is dialog mode.

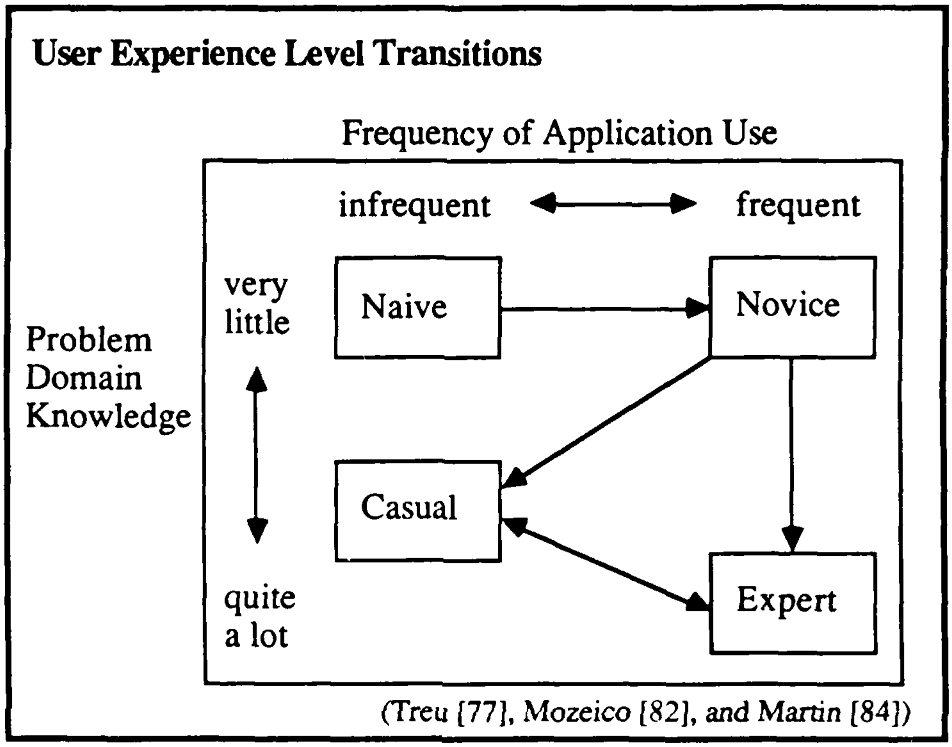
(1). Dialog Task: This dimension is constrained by the nature of the problem task and is not available for manipulation. The dialog task is determined by the nature of the problem task. It can not be changed to some other task to maximize performance or satisfaction even if user experience and dialog mode were fixed. This dimension can be, and usually is, multivalued since there may be more than one dialog task required in the interface to satisfy the problem task. This dimension is actually a constraint on the interface and does not provide the designer any latitude for improving performance or satisfaction. What does have an impact on the dialog task is the dialog mode used to accomplish the task and the underlying concepts used to support the accomplishment of the task.

(2). Dialog Mode: This dimension is critical to the interface since it is the actual method used by the user to communicate with the computer and is the only real independent variable at the designers disposal. The designer has the opportunity to select the dialog mode to maximize performance and satisfaction, given the constraints of the other dimensions. Restricting oneself to selecting only a single dialog mode ignores the differences in expert and novice problem solving and issues of memory and learning (the basis for the normative theory). Selecting a dialog mode without regard to the user population experience level could negatively impact performance and satisfaction especially if there is a miss-match of mode to experience. Thus, even though the designer can select the dialog mode, he should not make the decision arbitrarily and use mode to constrain the other dimensions. The dialog mode as well as the control of the dialog (user versus computer) must be made in concert with the other dimensions. An important research issue related to this that needs further work is determining the best dialog mode for a given dialog task when the user experience is a fixed value (e.g. only novices). While this is an important topic, this research narrows the scope of the user-system interface problem by restricting itself to just the relationship between user experience and dialog mode and does not consider which mode is best for which type of dialog task.

(3). User Experience: The ability to manipulate this dimension depends on the problem domain and the intended user population. The primary method for manipulating this dimension is to target all of the other dimensions to one level of user experience and force all users to migrate to the selected level through training and experience (Good, Whiteside, Wixon and Jones [84]). The main difficulty with targeting a dialog mode to a certain level of experience is that a user's level can undergo considerable change (Benbasat [84]). Larson [82] points out that not only can the current user population experience level changes, but other variables such as employee turnover can affect the experience level of the population.

Reflecting back to the experience level relationship diagrams shown in Figure 1.2 in Chapter I, this changing of experience would have the effect of continually increasing or decreasing the diameter of the circles and thus changing the relationships between technology, application and problem domain experience (the diameter of the circle represent the amount of experience). One of the most important user experience level transitions that can occur is at the intersection of technology and the problem domain; application experience. Use of an application can have an effect on both technology and problem domain experience levels and necessitate changes in the user-system interface. One can see then that trying to constrain user experience to one level of experience (a specific value in the model) for a constantly changing variable, and still maximize performance and satisfaction, is not realistic. An example of the type oi experience changes or transitions that can occur within application specific experience is depicted in Figure 2.3. These different transition stages or levels reflect prior research by Treu [77], Mozeico [82] and M. Martin [84]. Constraining this dimension, user experience, to one value may maximize performance and satisfaction for a subset of the user population at one of the levels depicted in the figure but it may also negatively impact the rest of the population at the other levels.

One solution to the range of value differences, especially for experience, is to have a completely adaptive interface. That is, one that conforms to each user's needs and changes as the user gains experience. Adaptive user-system interfaces can be thought of as having a stream of continuous dialog modes that can fit the user at whatever experience level necessary. While this appears to be the panacea for interfaces, it has not



(Treu [77], Mozeico [82], and Martin [84])

Figure 2.3

been proven so. Adaptive dialogs require some sort of underlying meta-knowledge of human behavior to understand his changing needs and adds a complexity to the interface that may not be warranted in many situations. Another method is to have only one dialog mode and let the user (or the system) adapt it to the user's requirements (Benbasat [84]). This method has the limitation of being only a user directed or computer directed dialog since there is only one mode. A less complex version of adaptive interfaces and one that ties in with Treu's and Mozeico's learning stages is to have a set of discrete dialog modes with each geared toward a certain level of experience. This simpler view of having multiple dialog modes is the one taken by this research.

A quote by H. Smith [80] sums up the problem with this dimension (user experience) and is a clear indication of why the researchers maintain that a user-system interface should accommodate different levels of user experience:

> The issue of designing computer systems for specific groups within the user community is complicated by the inherent adaptability of the human; today's tyro is tomorrow's informed user, or even expert. This transiency accounts for much of the difficulty of focussing on a particular category of user – they won't stay in their category long without changing their requirement.

(4). Underlying Concepts: A great deal of flexibility exists in this component or dimension of a user-system interface but these concepts apply to all dialog modes, task, and user experience levels. They are fundamental to the interface. Since extensive research has already been done in this dimension (though not necessarily implemented into design practices yet) this research assumes that they exist and are implemented in such a manner that any further manipulation would only decrease a user's satisfaction or performance.

(5). Conclusion: From this discussion of the four dimensions, it can be concluded that the key to supporting both novices and experts in a user-system interface, while maximizing performance and satisfaction, lies with the interaction between the user experience level and the dialog mode and that a viable solution is to have multiple dialog modes. This conclusion is made under the assumption that the interface contains the necessary fundamental underlying concepts and that the user population experience level is varied.

Experimental Support

Given the theoretical and conceptual support for accommodating user experience level in an interface, the question that needs to be answered experimentally is: "Is the theory true and will multiple dialogs improve satisfaction and performance for a user population that has different levels of experience?" Most of the previous experiments done concerning user-system interfaces have dealt with some underlying concept in relationship to novices only or with experts versus novices. Few experiments have tried to collect experimental evidence for the need to provide multiple dialog modes for accommodating the differences in user experiences as the before mentioned normative theory has suggested. Those few experiments that did deal directly with this issue failed to provide any support for this theory but in each case the experiment had serious internal validity problems or confounding variables that cause one to be suspect of the results. There are also a few other experiments that do provide support for multiple dialogs but the data was collected as a by-product of other research and was not the primary focus; thus it can not necessarily be generalized beyond the specific instance.

One experiment that provided support for this theory was done by Hiltz [84]. She was studying the effects of implementing an electronic mail system into an organization of 100+ scientists who were naive on the use and functions of electronic mail. While studying the changes in user behavior and attitudes toward the system over time, she found strong evidence that the users' behavior and preference changes (that were related to features or capabilities in the system) were a function of level of experience (hours on-line). The most universally appreciated feature (satisfaction element) was the ability to use system commands to replace a menu driver interface after the users understood the options available. Another perceived useful function that increased with experience was: "Facilities that allow a user to actively control the system rather than passively react to menu choices...." [p.g. 105]. These particular observations lend support to the contention that the user's choice of user directed versus computer directed dialogs is a function of user experience. While Hiltz looked at satisfaction and preference behavior, she did not consider performance. She also did not do a pretest to determine technology experience. Her data was based on a 6 month and 18 month user survey but she had a problem in that the subjects were not required to participate in the survey or even use the system. Her results were biased toward those that used the system the most, especially beyond the 6 month point. So while her data supported the theory, there needed to be more control over the subjects. In particular, what was the reaction of those subjects that only used the system occasionally and never progressed beyond novice or casual.

Another study that supports Hiltz's findings on users switching from computer directed to user directed dialogs was found by Gilfoil [84]. He was primarily interested in investigating the cognitive knowledge structure that a user built and changed as he gained experience with the interface and its command language. As part of the study he also wanted to determine at what level of experience users would switch from computer directed dialog to user directed dialog to see if the knowledge structure changed at the same time. In his study, he used four novices who performed 30 different tasks over a one month period. The interface consisted of a computer initiated question and answer dialog mode and a user directed command language mode. He found that all four of the subjects switched to the user directed method by the time they had had 15 to 18 hours of experience wit'ı the system. Interestingly enough, one of the users prefered to use both of the dialog methods for the remainder of the experiment (depending on the specific task he needed to do) because it was easier for him that way (satisfaction). Gilfoil also obtained performance data but it was only for ensuring that performance improved with experience, which it did. While these results support the normative theory, they are limited in their generalizability since only four subjects were used.

Benbasat, Dexter and Masulis [81] conducted a study primarily to analyze whether the interface (computer directed versus user directed) and user characteristics (experience, cognitive style, and risk taking) affected decision effectiveness and subject behavior in an interactive problem solving situation. Their dependent variables were performance and use of system options. The results that are important to this study were the experimenters' observations (but not statistically significant) that novices had much more difficulty with a user directed dialog (commands) then did those with computer directed dialog (question and answer). Experts were unaffected. They felt significance would have been achieved on this difference if both of the dialogs were not so simplistic and a more rich command structure was used. One extremely important contribution made from this research was the development of a framework to investigate human/computer interaction. This framework will be discussed more fully in methodology chapter and was the basis for the experiment conducted in this research.

Mozeico [82], to show that his learning stages did have a direct relationship to the dialog mode, built an interface to a graphics system to accommodate the different learning stages. Using novices in technology and problem domain, he obtained some positive results to support his theory. The problem with his experiment was that it was more of a demonstration than a rigorous experiment. He also found that going from a question and answer (computer directed) dialog mode to a command language (user directed) as one transitioned from the novice to the experienced level was too large a difference in modes. He suggested that menus be used to ease the transition.

Hauptmann and Green [83] hypothesized that a natural language (user directed) dialog mode for an interactive graphics system would allow subjects to perform better and be more satisfied than subjects who used a menu or a command language dialog mode. They had subjects draw a pie, bar and line chart as the task using one of three interfaces. The order of the tasks were varied within each treatment group. Their results showed no significant difference between dialog modes for the two dependent variables. What did make a difference was the ordering of the tasks. Those who did a pie chart last had higher performance and satisfaction (regardless of the dialog mode) than any other ordering. The researchers concluded that this intervening variable was the cause for no significance. They also had internal validity problems with subjects in that they did not control for technology experience (James [84]) in subject selection or assignment to treatment groups. A second problem was the measurement of performance. Several of the measurements were biased unfavorably toward one or two of the dialog modes.

Whiteside, Jones, Levy and Wixon [85] conducted an experiment in which subjects were required to perform file manipulation activities using one of seven different operating systems (1 menu, 2 icon, and 4 command language). Subjects were categorized as new users, transfer users (experienced but not with the treatment interface) and system users (experienced with the treatment interface). While they found the expected performance differences between subjects classifications, there was no difference in performance or satisfaction between systems. All classes of users did best with one of the command language interface and all classes did worst with the menu interface, with one of the icon interfaces comming in a close second. This result is inconsistent with not only the proposed normative theory and theoretical support discussed earlier but with other research and thus needs to be carefully examined. It was found that there were two major confounding elements in this experiment that contributed greatly to their findings. The first problem was that the researchers used seven different computer operating systems running on several different types of computer hardware. Measurement of performance across different operating systems and hardware using "time to complete task" as part of the measurement for performance ignores the tremendous differences that can occur due to differences in underlying concepts and implementation differences in file operations. The researchers themselves acknowledged this problem in a back handed way by making conclusions about the importance of underlying concepts. They said [pg 190]: "...the care with which an interface is crafted [underlying concepts and dialog mode specifics] is more important than the style of interface [dialog mode] chosen...This...care probably comes with product maturity...[but that] process can be accelerated by careful application of human factors theory." These problems with the basic underlying concepts in some of the interfaces used in the experiment were the second major problem.

The conclusion to be drawn then is that a well designed and controlled experiment that alleviates the problems noted in these previous experiments is still needed to test whether or not multiple dialog modes can support the theory and are better than a single dialog mode when there is a range of user experience in the population.

Summary

From the body of literature on interface components, differences in expert and novice, and the cognitive factors of memory and learning, it is evident that designing an interface to facilitate high user performance and satisfaction is a non-trivial task. The literature has shown that dialog mode, dialog task, user experience and underlying concepts all play an important roll in the usability of the interface to achieve the accomplishment of a task. It has been shown that while these four elements make up the components of an interface, differences in expert and novice problem solving and the cognitive issues of memory and learning must be considered in concert with these four elements. An important question that arose from these interactions and which was addressed is how all these elements interact with each other and which ones have the most effect on achieving the goal of maximizing user performance and satisfaction in accomplishing a task. It was shown that the dialog mode component was the only real candidate for being manipulated within a user-system interface for accommodating different levels of user experience (without ignoring the cognitive processes) and still achieve the interface goals. This chapter concluded with a review of the relevant prior experiments and showed how some supported the theory whereas those that did not had validity problems. From this information, an experiment was designed and conducted to answer the research and investigative questions first set out in Chapter I and supported here in Chapter II. The following chapters will discuss the research methodology, experiment, results, and the conclusions drawn from the experiment.

Chapter III
Research Methodology

This chapter describes the methodology and the experimental design used for this research. Prior to discussing the actual experiment, the specific hypotheses developed from the research and the investigative questions presented earlier are addressed. To re-iterate the research questions:

1. Will the availability of multiple dialog modes in a user-system interface that accommodates different levels of user experience make a difference in overall user performance and satisfaction?

2. At what level of experience will a user prefer a user directed dialog over a computer directed dialog?

Hypotheses

As a consequence of the research and investigative questions posed, the following are the specific hypotheses examined in this study. The hypotheses are stated in three parts: (1) those that apply to performance only, (2) those that apply to satisfaction only and (3) those that apply to a user's application experience level.

Hypotheses on Performance: The first hypothesis (H1) tests to see if multiple dialog modes make a difference on performance. Hypotheses H2 - H4 all state that for a mixed population, experts only, and novices only, in that order, multiple dialog modes lead to better performance.

$H1_0$: An interface with multiple dialog modes (one computer directed and one user directed) has no effect on a user population's performance over an interface with one dialog mode.

$H1_A$: There is a difference in a user population's performance between an interface with only one dialog mode versus an interface with multiple dialog modes.

$H2_0$: Subjects who have multiple dialog modes (both computer and user directed) do not perform as well as users who only have a single dialog mode (either computer directed or user directed).

$H2_A$:Subjects who have multiple dialog modes perform better than users who only have either a computer directed or user directed dialog mode.

$H3_0$: Experts users who have multiple dialog modes (both computer and user directed) do not perform as well as those experts who only have a single dialog mode (either computer directed or user directed).

$H3_A$: Experts who have multiple dialog modes perform better than experts who only have either a computer directed or user directed dialog mode.

$H4_0$: Novices users who have multiple dialog modes (both computer and user directed) do not perform as well as those novices who only have a single dialog mode (either computer directed or user directed).

$H4_A$: Novices who have multiple dialog modes perform better than novices who only have either a computer directed or user directed dialog mode.

Hypotheses on Satisfaction: These four hypotheses are similar in content and format to the performance ones except that they apply to the user's satisfaction with having multiple dialog modes.

$H5_0$: An interface's dialog modes (computer directed, user directed or both) has no effect on a user population's satisfaction with the interface.

$H5_A$: An interface's dialog modes (computer directed, user directed or both) has an effect on a user population's satisfaction with the interface.

$H6_0$: Subjects who have multiple dialog modes (computer and user directed mode) were equally or less satisfied with the user-system interface than users who only have either a computer directed or user directed dialog mode.

$H6_A$: Subjects who have multiple dialog modes are more satisfied with the user-system interface than users who only have either a computer directed or user directed dialog mode.

$H7_0$: Experts who have multiple dialog modes (computer and user directed mode) were equally or less satisfied with the user-system interface than experts who only have either a computer directed or user directed dialog mode.

$H7_A$: Experts who have multiple dialog modes are more satisfied with the user-system interface than experts who only have either a computer directed or user directed dialog mode.

$H8_0$: Novices who have multiple dialog modes (computer and user directed mode) were equally or less satisfied with the user-system interface than novices who only have either a computer directed or user directed dialog mode.

$H8_A$: Novices who have multiple dialog modes are more satisfied with the user-system interface than experts who only have either a computer directed or user directed dialog mode.

Application Specific Hypotheses:

$H9_0$: Subjects with more that 18 hours of application specific experience will be equally or more satisfied with only a computer directed dialog mode (menu) than those who have both or only a user directed.

$H9_A$: Subjects with more that 18 hours of application specific experience will be less satisfied with only a computer directed dialog mode (menu) than those who have both or only a user directed.

$H10_0$: Subjects with more than 18 hours of application specific experience will not use the user directed mode more than the computer directed dialog mode.

$H10_A$: Subjects with more than 18 hours of application specific experience will use the user directed mode more than the computer directed dialog mode.

Experimental Design

This research used volunteer subjects in a controlled laboratory setting performing tasks with different types of dialog modes. Selection of the independent and dependent variables for this experiment are supported by the prior work of Benbasat et al [81] (Figure 3.1) who established a model for conducting user-system research. The italicized items in the figure are those independent and dependent variables chosen for this study. Specifically, the independent variables are the dialog mode(s) used to perform a task and a subject's technology experience.

The dialog mode variable consists of three treatments: (a) menu only (computer directed), (b) command language only (user directed)

Framework for Investigating the Human/Computer Interface

(Benbasat et al, 81)

Independent Variables				Dependent Variables
Human user	Decision Environment	Task	Interface Characteristics	Human/ Computer Effectiveness
Demograhpics age education ***experience*** Psychological cognitive style intelligence risk attitude	Decision Structure Organizational Level Other Characteristics stability time pressure uncertainty	Decision Support Inquiry Information Retrieval CAI Word Processing Data entry	Input/Output Media ***Dialog type*** Language Characteristics Presentation Format	Performance ***time*** ***errors*** ***task completion*** profit User attitudes ***satisfaction*** confidence Use of System Options

Figure 3.1

and (c) both menu and command language together. The technology experience variable has two classes determined by a pre-survey of potential subjects: (a) novice and (b) experienced (hereafter called experts in this research).

To ensure internal validity for this experiment, the other three categories of user experience (discussed earlier in chapter 2) were also controlled. Problem domain experience was held constant by selecting an easily understood task with which all subjects would be familiar (updating an address book). General experience was assumed to be constant since all subject were university students and college graduates. Application specific experience could not be assumed constant or ignored as with the other two categories without posing possible internal validity problems plus it was an item of interest to the research. To completely control for both technology and application experience, the type of interaction shown in Figure 3.2 needed to be considered but it posed subject selection problems and unnecessarily complicated the experimental design. As an alternative to this design, technology was selected as the independent variable but all subjects were measured on their application experience to be investigated as a covariant in the experiment. Application experience was also used to further divide the sample population for random assignment to the treatment groups.

The dependent variables for this experiment are performance effectiveness/efficiency and user satisfaction with the dialog mode(s) used to do the tasks. Performance was measured by: (a) time to complete the task (efficiency), (b) quantitative score reflecting correctness of

User Experience Levels

		Technology	
		Expert	Novice
Application (dBASE)	Expert	1	2
	Novice	3	4

Figure 3.2

required actions (effectiveness) and (c) % menu used vs command language when a choice was available. The satisfaction measurement was accomplished with a semantic differential post-survey given to each subject at the conclusion of the experiment. The survey is based on similar survey instruments developed and tested by Zoltan and Chapanis [82], Baily and Pleason [83], Magers [83], Hauptman [82] et al, and Kerber [83].

Figure 3.3 relates these independent and dependent variables to the conceptual model presented in an earlier chapter. To restrict the scope of the experiment and ensure validity, the dialog task elements were collectively contained in each of the experimental tasks and not tested individually. Other researchers might find it interesting to consider them as an independent variable in some other experimental setting. The underlying concepts dimension (after a careful examination) was assumed to be acceptable in the application used for the experiment.

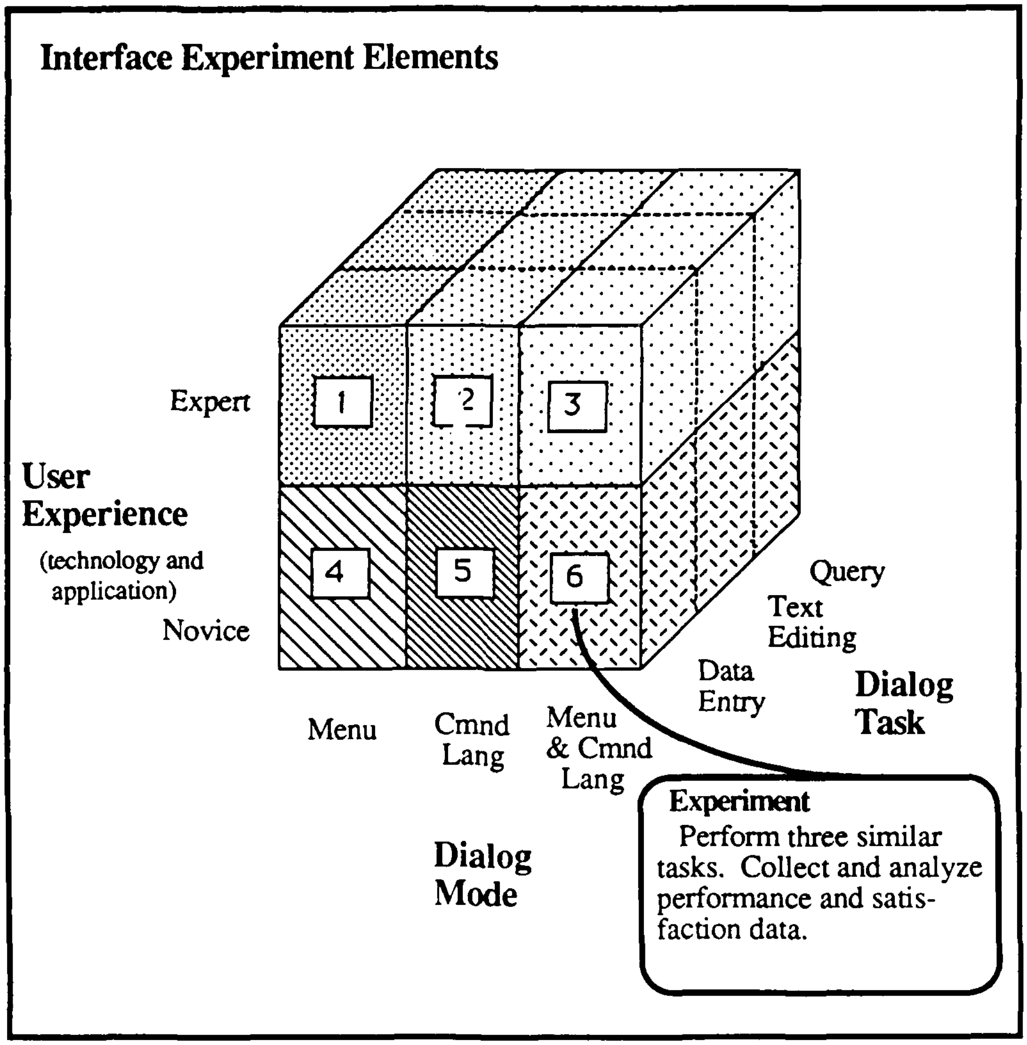

Interface Experiment Elements
1
2
3
4
5
6
Expert
User Experience
(technology and application)
Novice
Menu
Cmnd Lang
Menu & Cmnd Lang
Dialog Mode
Query
Text Editing
Data Entry
Dialog Task
Experiment
Perform three similar tasks. Collect and analyze performance and satisfaction data.

Figure 3.3

The experimental design can be better described in a more traditional two dimensional diagram as shown in Figure 3.4. This shows that the experiment has three treatment groups with a blocking factor of two for a total of 6 different experimental cells. Each subject was assigned to only one cell and was required to perform two specific tasks consisting of multiple steps or items; thus, for performance, there is a

Experimental Design

		Dialog Mode		
		Menu	Command Language	Menu & Command Language
User Experience	Expert	1 performance satisfaction	2 performance satisfaction	3 performance satisfaction dialog usage
	Novice	4 performance satisfaction	5 performance satisfaction	6 performance satisfaction dialog usage

Figure 3.4

third dimension of two entries (not shown) that represents the two tasks. The contents of each cell lists the dependent variables.

To control for extraneous and confounding variables: (a) the same two tasks were used for every treatment group, (b) the tasks were accomplished using a software package that has the facility to provide multiple dialog modes (this eliminated discrepancies in application performance and response times that could occur if different packages were used), (c) every group used the same type of hardware and operating system and (d) every group performed the tasks in the same setting. The following section describes the details of the experiment.

Task/Treatment

Task Design: The specific application used for the experiment was dBASE III. It was selected over other applications because of its rich command language, the ability to develop menus and the ability to surreptitiously generate a log of all console actions and system responses. The two dialog modes tested were a restricted subset of the dBASE III command language and a menu system that was developed using the dBASE III macro and screen generation facility (see Appendix A for a list of the commands used in the experiment). The menu system was built using proven design techniques and contained a sufficient set of the underlying concepts described in Chapter II as necessary for any type of interface dialog mode. Extensive care was taken in developing the menu interface to ensure that it was not excessively better or worse (in terms of human factors issues and underlying concepts) than the built in dBASE III command language. The capabilities of the two dialog modes were made equivalent so that even though the menus were application specific to the address book task and used terms applicable to the task, anything that could be done with the restricted subset of commands from the command language could be done with the menu system and vice versa. To align the two modes even more, the menu system presented information and asked for information using terms and formats consistent with the command language. All of this was done to ensure that there would not be an inherent bias toward one or the other of the modes. Pilot testing of the system indicated that this was successfully accomplished. Appendix B presents a detailed description of the menu system.

The problem tasks themselves consisted of 14 numbered items (8 in task one and 6 in task two) that required the use of different dialog tasks such as adding, deleting and modifying data records (data entry and text editing tasks), querying data base, and generating reports (querying and data entry tasks). This mixture of dialog tasks ensured that the problem tasks were not biased toward a particular dialog mode because only one dialog task was used. Appendix C contains the two tasks given to the subjects plus additional information on scoring the tasks for generating the performance measures.

Subjects: Subjects were primarily undergradua e and graduate students from the University of Texas College and Graduate School of Business. Volunteers were solicited from a graduate class that presented an introduction to Information Systems (64 volunteers), the undergraduate COBOL programming classes (66 volunteers) and other PhD students and professionals (15 volunteers). Subjects from the graduate and undergraduate classes were offered bonus points toward their final grade as an incentive for participating in the experiment (although many of the subjects indicated to both the experimenter and the classroom instructors that their primary motivation for participation was driven by their interest in the experiment and not the points). All subjects that volunteered completed a Computer Technology Experience Survey (see Appendix D) that was used to categorize potential subjects as experts or novices in computer technology and as expert, novice or no experience with the dBASE III system. The primary factors used to determine a subject's experience level were: (a) course work, (b) work experience, (c) programming languages used, (d) hardware used and (e) types of applications used. A copy of the survey can be found in

Appendix D along with a detailed description of the scoring technique and score distribution. An analysis of the scores derived from a presurvey of subjects that were known to be novices or experts held true for the whole population sampled with the survey. Novices grouped around a score of 10 to 24 while experts had scores greater than 30. Those who scored between 24 and 30 were re-analyzed with the cut off for novice being 26 (the survey indicated a clear break at this value). The mean experience score for experts was 44.5 and for novices15.3. From this initial population of 148 volunteers, a total of 98 subjects participated. Based on the subjects' technology and application specific experience as determined by the computer technology survey, they were assigned to either the novice or expert group and then either randomly selected from these two groups to one of the three treatment groups or selfselected to a group without knowledge of their experience rating or treatment to be received. Figure 3.5 summarizes the specifics on the distribution of subjects to each treatment based on their technology and dBASE experience level.

All subjects had at least some experience with computer technology and thus did not require any further training in this area. Most of the subjects had at least some training with the dBASE command language gained by either course work in school or work experience. Those without any experience with dBASE received (along with all other subjects) a short introductory training as part of the experiment.

Procedure: There was a total of 12 experiments conducted on different days of the week at either 5 to 7 pm or 7 to 9 pm. A thirteenth

Subject Assignment

		Dialog Mode			
		Menu	Cmnd Lang	Menu & Cmnd Lang	Total
User Experience Level : technology - application	Expert-E	6	3	6	15
	Expert-N	3	4	5	12
	Expert-Z	4	4	3	11
	Total	13	11	14	38
	Novice-E	1	1	1	3
	Novice-N	13	13	12	38
	Novice-Z	6	7	6	19
	Total	20	21	19	60
				Total	98

Legend:
E - Expert in dBASE
N - Novice in dBASE
Z - No experience in dBASE

Figure 3.5

experiment was run on a Saturday morning from 9 to 11 am. The experiments were conducted at these time frames to reduce scheduling conflicts with both the subjects and the laboratory used for the experiment. The size of the experimental group varried from a low of 4 subjects to a maximum of 11. All of the experiments were conducted in the same room using IBM PC equipment. The room was isolated from all other areas so that only the subjects and the monitor were in the room during the experimental session.

At the beginning of each experimental session, subjects were given a 30 minute training session on the particular dialog mode(s) that they would be using. This training was done in a group setting using an overhead projector and handouts (see appendices A and B for copies of the handouts and documentation given to subjects). The training consisted of an introduction to the purpose of the experimert, an overview of the problem domain and associated data base formats, a presentation of the commands or menus that were in the dialog mode the subject would be using and a short hands-on session prior to begining the first task. For those subjects that had both menu and command language at their disposal for the experiment, the training session was about 15 minutes longer since two modes had to be discussed. In all cases, regardless of their previous experience, all subjects received the same training for their dialog mode.

The problem task the subjects were asked to perform consisted of updating an address book and an associated Christmas card list. This type of task was selected to ensure that all of the subjects would have the same problem domain knowledge and thus eliminate this variability. In order to accurately collect performance data and a user's satisfaction with the dialog mode(s), the experimental task had to be fairly lengthy and contain repetitive types of action so the subjects could become familiar with the interface mode(s) . To accomplish this, the experimental task, presented in the form of a short narratives describing what needed to be accomplished (see Appendix C for a list of the items in each task), was split into two 45 minute parts so that the time limits would not be perceived as too long and so that there would be a sense of closure (Sneiderman [80]) on finishing one part before going on to the next. The time limit was imposed so subjects would have some pressure to perform the items not only correctly but in a timely manner. There was a two to three minute break (although subjects were not allowed to leave the room) between the two tasks. During each of the 45 minute time periods, the subjects were asked to complete as many items as possible from the task. The order in which the tasks were given to the subjects was not varied between subjects or treatments. The subjects were allowed to ask for clarification on item requirements and for help in understanding information from the handouts since the purpose of the experiment was for analyzing performance and satisfaction using specific dialog modes and not interpretation of the item requirements or documentation. Subjects who had questions that pertained to the dialog mode's functions or syntax were directed to the documentation to find the answer.

When each subject finished the second task or when the time limit was reached, they were given a Satisfaction Survey (see Appendix E) that contained seven semantic differential pairs of words from which a satisfaction score was derived. Other questions also were asked to gather information on their thoughts on the experiment itself and interface dialog modes in general. There was a comment section at the end where subjects were encouraged to express any additional feelings they might have on dialog modes and the experiment itself that were not specifically addressed in the survey questions. Subjects could leave the laboratory once the survey was complete.

Analysis Procedures

Data Gathering: To determine the performance measurement variable, each subject's actions and the system's responses were collected in a log file. From a printed copy of the log (see Appendix F for examples), a quantitative score was determined for each task by analyzing each item in the tasks and assigning it a score based on a scale of 0 to 4 where a 0 was given for not even attempting to do an item and a 4 was awarded for a completely correct action/response. To determine what constituted a score of 1 to 3, five logs from each mode were analyzed for establishing partial credit on each item. Once the scoring standard was set, it was only modified to include new situations not found in the logs used to set the initial criteria. To ensure that the scoring technique did not unintentionally add a bias to a subset of the subjects since the interval range of 0 to 4 was subjectively determined, alternative scoring methods were analyzed (Emory [80]). From this analysis it was concluded that the difference between a score of 0 and 1 was negligible and very few scores of 1 were given (e.g. the range was really 0,2,3,4). Thus , all item scores of zero were converted to a 1 for a final range of 1 to 4. The maximum score then was 4 times the number of scored items which was 13 for task 1 and 10 for task 2. While there were only 14 numbered items in total, some items had multiple parts that were scored separately. Thus task 1 had a maximum score of 52 and task 2 a 40.

The efficiency variable was determined by time stamps that were put into the log by the recording function. There was a time score generated for each task (though not for each item) with the maximum value being the 45 minute task time limit. The time score was then divided by the number of items in the task with a score greater than 1 to generate an average time per task.

The satisfaction variable was generated by adding up the score for each of the seven items used to elicit the subject's satisfaction then dividing by seven to get a score that ranged from 1 (extremely satisfied) to 7 (extremely dissatisfied).

Statistics: The analysis of the data was done using the SAS statistical package on an IBM 3081. Both the performance and satisfaction variables were analyzed using an analysis of variance for unbalanced cells. The model used for the analysis of the performance variables is:

Performance = A B C D(AxB) AxB

where the class variables are defined to be:

A -> TECHEXP - technology experience (expert or novice)
B -> TREAT - dialog mode treatment (menu, command language,both)
C -> TASK - experimental tasks
D -> ID - id of subject (a unique value)

The subject (ID) is nested in the TECHEXPxTREAT interaction and has a repeated measure on TASK.

The satisfaction model is simpler since there was only one measurement per subject:

Satisfaction = A B AxB

Both the performance and satisfaction models were also analyzed using dBASE experience as a covariant to determine the effects of application specific experience on these variables. Other statistics such as comparison tests, correlations and means were used as needed to investigate each of the hypotheses.

Chapter IV
Results and Discussion

The first section of this chapter discusses some observations and results obtained from the experiment that have to do with the validity of the research. Experimental results are then discussed in the next three sections and correspond to the three major categories of hypotheses: performance, satisfaction and application specific knowledge. One additional section is included to present additional findings that apply only to those subjects in the treatment that had both menus and command language. A final section presents an overall discussion of the experimental results. Conclusions that can be drawn from this research as well as potential future research areas that can be built upon this research are presented in the next chapter.

Several different items were measured as part of the experiment to ensure that there was not some confounding variable inherent within the experiment that caused validity problems with the measurement variables. In particular, all the subjects were asked to rate their satisfaction with the tasks they were required to perform and their overall satisfaction with the experiment itself. Low satisfaction scores on either of these items by the whole subject population or by a specific subset could indicate some sort of problem with the experiment itself and confounded the results. In this case, there were no indications of any problems. The means for the two questions were 2.6 for task satisfaction and 2.8 for experiment satisfaction for the population as a whole (1=extremely satisfied, 4=neutral, 7=extremely dissatisfied). There was no significant deviation from these scores for any specific subgroup of subjects. The conclusion drawn then is that the measured performance and satisfaction variables were not biased by ill feelings toward the experiment itself and reflect their real performance and actual feelings as elicited by the survey instrument.

Performance Hypotheses

It was predicted that experienced users would perform better with a user directed dialog mode (command language) than those who used a computer directed dialog mode (menu). This was based primarily on the differences in expert versus novice problem solving in that the command language would compliment the expert's problem solving strategy. The opposite was predicted for novices for the same reasoning; e.g. menus complimented the novice's strategy. Therefore, when one considers a mixed user population of novices and experts, their performance in accomplishing a task using a user-system interface with only a computer directed or a user directed dialog mode should not be as good as an interface that has both a computer and user directed dialog mode.

The following hypotheses test this prediction with H1 just looking at whether or not multiple dialog modes can affect performance. Given there is an effect, H2, H3, and H4 evaluate the specific performance of the whole population, experts only, and novices only across the three treatments.

Hypothesis 1

$H1_0$: An interface with multiple dialog modes (one computer directed and one user directed) has no effect on a user population's performance over an interface with one dialog mode.

$H1_A$: There is a difference in a user population's performance between an interface with only one dialog mode versus an interface with multiple dialog modes.

Hypothesis 2

$H2_0$: Subjects who have multiple dialog modes (both computer and user directed) do not perform as well as users who only have a single dialog mode (either computer directed or user directed).

$H2_A$: Subjects who have multiple dialog modes perform better than users who only have either a computer directed or user directed dialog mode.

H3 and H4 are basically identical to H2 except that they apply, respectfully, to experts only and novices only as opposed to the whole population.

Hypothesis 3

$H3_0$: Experts users who have multiple dialog modes (both computer and user directed) do not perform as well as those experts who only have a single dialog mode (either computer directed or user directed).

$H3_A$: Experts who have multiple dialog modes perform better than experts who only have either a computer directed or user directed dialog mode.

Hypothesis 4

$H4_0$: Novices users who have multiple dialog modes (both computer and user directed) do not perform as well as those novices who only have a single dialog mode (either computer directed or user directed).

$H4_A$: Novices who have multiple dialog modes perform better than novices who only have either a computer directed or user directed dialog mode.

Performance was determined by two factors: (1) effectiveness (average score per item in each of the two tasks) and (2) efficiency (average time to complete each item in each of the two tasks). H1 through H4 are evaluated separately on both of these factors with effectiveness being first .

Results for Effectiveness

In this analysis, an average score of 4.0 is the maximum possible value and constitutes a perfect effectiveness score. The minimum score is 1.0. Figure 4.1 presents the combined task 1 and 2 effectiveness mean score for experts only, for novices only, and for the entire population. A simple inspection of the figure clearly indicates that there was a difference in performance for the different interfaces.

Testing of H1: The test for statistical significance of the differences in the means for performance effectiveness was done using ANOVA with the results depicted in Table 4.1. The associated comparisons between the three dialog mode treatments for a mixed population of experts and novices are shown in Table 4.2. The results in Table 4.1 clearly indicate

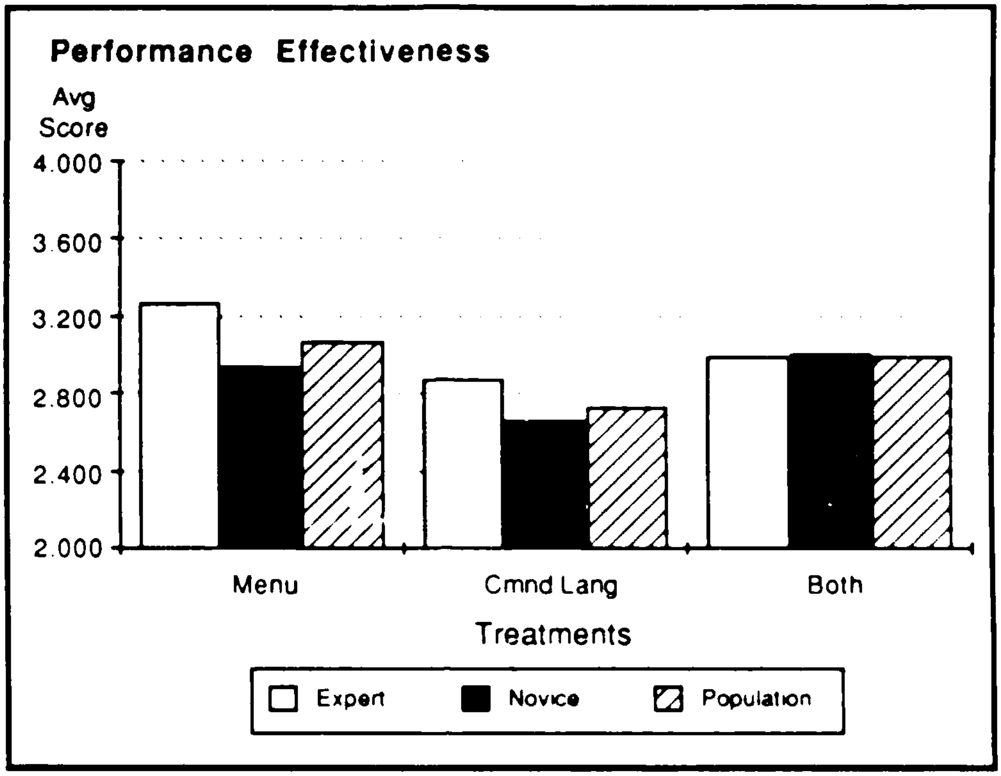


Figure 4.1

Performance Model - Effectiveness

Source	Df	SS	F Value	p>F
Model	93	48.327	4.85	<.0001
Error	92	9.862	N/A	N/A
TECHEXP	1	1.379	12.86	<.0005
TREAT	2	3.316	15.47	<.0001
TASK	1	0.948	8.84	<.0038
TECHEXPxTREAT	2	0.845	3.94	<.0227
ID(TECHEXPxTREAT)	87	41.262	4.42	<.0001
MSE = .107 R^2 = .63				

Table 4.1

significance (p<.0001) for the model and for the different dialog mode treatments (TREAT variable). Thus $H1_0$ is rejected and the alternate hypothesis is accepted meaning there was a difference in performance based upon the interface and dialog modes used.

Testing of H2: The preferred method for the analysis of effectiveness for H2 and the next two hypotheses is to use a combined average value for the task 1 and 2 effectiveness scores and not evaluate each task separately. To do this it was first necessary to see if there is any significant difference in the mean scores between the two tasks. The means for the two task scores were plotted for experts and novices by treatment (see Figure 4.2). This chart shows that while the pattern of means between the two tasks is the same, task 2 means are greater than task 1. To see if this was a significant difference, a comparison was done between task 1 and task 2. Lack of significance would allow further analysis to use the combined scores from both tasks. Significance could indicate that other sources of variation had entered the experiment such as one task significantly more difficult than another, user fatigue, etc. and would force a separate analysis for each task. It was expected that the mean for the scores in task 2 would be slightly higher than task 1, due to learning, which it was (task 1 overall effectiveness mean = 2.86, task 2 overall effectiveness mean = 3.00).

To do the comparison test between the two tasks, it was necessary to use MSwithin subjects (42.26/87) as the error term since the variable TASK is a repeated measure within ID (TECHEXP x TREAT)

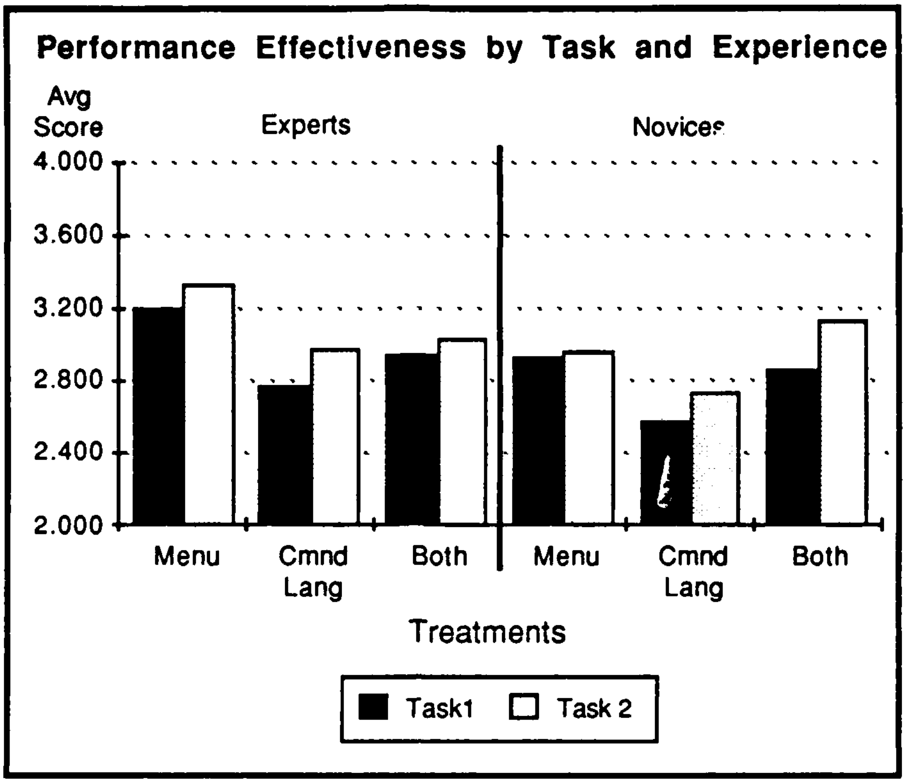


Figure 4.2

(Winer [71]). The comparison test showed no significance (SS = .891, F = 1.879, p>.05) and therefore the task scores are combined in the analysis and are shown in Table 4.2.

From Table 4.2, it can be seen that while the mean effectiveness score for the menu dialog mode treatment was actually greater than the combined menu and command language dialog mode treatment (hereafter referred to as "both"), it was not significant. There was significance at the p<.01 for "both" compared to command language as predicted. The menu treatment was also significantly better than the command language treatment. Since the "both" treatment was not better than both of the other two treatments $H2_0$ cannot be rejected.

Treatment Comparisons using Total Population: Performance Effectiveness

Comparisons		Means		F Value	p>F
(a)	(b)	(a)	(b)		
"Both"	Menu	2.996	3.067	3.290	<.10
"Both"	Cmnd Lang	2.996	2.729	13.677	<.001
Menu	Cmnd Lang	3.067	2.729	29.595	<.001

MSE = .107 F(.01) = 6.96 F(.05) = 3.96 Df = 1/92

Table 4.2

Testing of H3: Table 4.3 indicates the results of comparing the "both" treatment to menu and command language for those users classified as experts. In this case, the menu dialog mode was significantly better ($p<.01$) then either the "both" or the command language dialog modes; thus, $H3_0$ is not rejected. The mean for "both" was greater than for command language but not significantly.

Testing of H4: For novices the mean for "both" was greater than the other two treatments (see Table 4.4) but there was only significance ($p<.01$) for "both" to command language. There was no significance between "both" and menu thus $H4_0$ could not be rejected.

Discussion: From the analysis, it has been shown that for a mixed population and for novices, there was no statistical difference between the effectiveness score in the "both" treatment and the menu treatment and there was a statistical difference between "both" and command languages. For experts it would not be totally unexpected if there was no difference in effectiveness scores between treatments since both

Treatment Comparisons using Experts : Performance Effectiveness

Comparisons		Means		F Value	p>F
(a)	(b)	(a)	(b)		
"Both"	Menu	2.992	3.265	8.710	<.005
"Both"	Cmnd Lang	2.992	2.878	1.355	>.10
Menu	Cmnd Lang	3.265	2.878	15.237	<.001
MSE = .107 F(.01) = 6.96 F(.05) = 3.96 Df = 1/92					

Table 4.3

Treatment Comparisons using Novices : Performance Effectiveness

Comparisons		Means		F Value	p>F
(a)	(b)	(a)	(b)		
"Both"	Menu	2.998	2.943	.542	>.10
"Both"	Cmnd Lang	2.998	2.658	20.925	<.001
Menu	Cmnd Lang	2.943	2.658	15.047	<.001
MSE = .107 F(.01) = 6.96 F(.05) = 3.96 Df = 1/92					

Table 4.4

command language and menu are familiar interface methods and most of the experts had used dBASE before. This was not the case though. Menu turned out to have a far better score than either of the other two methods while the other two were statistically equivalent (as could be expected if "both" was not better).

As part of the overall analysis of effectiveness, it is important to know whether or not the amount of experience the subject had with dBASE influenced the effectiveness score as a co-variant with technical experience. Figure 4.3 displays the mean score for three classifications of dBASE experience (expert, novice, and no experience) by technical experience and treatment. To simplify terminology in the figures and discussion, the format for identifying each of the subject subgroups will be: [technical experience] / [dBASE experience] where the experience level is identified by (1) E - expert, (2) N - novice and (3) Z - no experience (e.g. E/E means expert in both technology and dBASE).

For experts/dBASE novices (E/N), the score fluctuates much more than the other two categories and most likely reflects the difficulty in distinguishing between experts and novices in an application and the wide range of technical experience differences in experts. The co-variant analysis of dBASE experience on effectiveness for these treatments showed no significance (SS=.0205, F=.191, p>.1).

The novices without dBASE experience (N/Z) were less effective in the command language and "both" treatments but in the menu treatment they actually had a slightly higher mean than those novices with dBASE experience. The basic equality of scores between the two novice

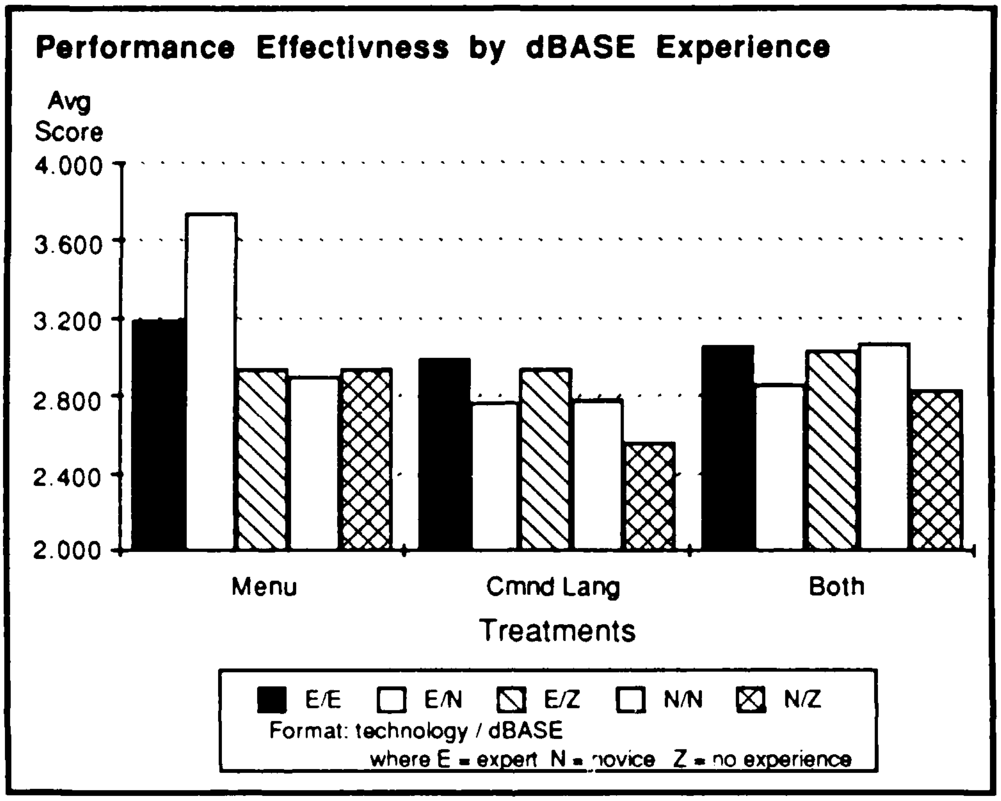


Figure 4.3

groups demonstrates one of the benefits of menu interfaces for novices. The use of menus allowed novices without any dBASE experience (and in some cases no data base experiences at all) to score equally as well as those novices with dBASE experience. The last item to notice about performance effectiveness is that the N/N group in the both treatment scored higher than all groups except the E/E and E/N groups in the menu treatment and they obtained this high score with 30 percent of the subjects in the group using command language 50% or more of the time.

The experiment did not prove that in this setting, an interface with multiple dialog modes produces more effective results for either a mixed or specific population than one with just one dialog mode although they can be statistically equal. The question of whether or not 'equal' is a worthwhile pursuit has to be determined by the interfaue developer.

Results for Efficiency

The efficiency score reflects the average number of minutes it took a subject to accomplish an item in each of the tasks. The lower the score the more efficient the subject or population was in accomplishing each item in the task. The range for efficiency scores went from 1.44 to 22.5 minutes per item in a task. Figure 4.4 shows that there was definitely a difference in efficiency scores between treatments for both experts, novices and the combined population.

Testing of H1: The results of the ANOVA on the efficiency variable are very similar to the performance results. Efficiency was also significant at $p<.0001$ level (see Table 4.5) for the model and the dialog mode treatments (TREAT). There was no significance in the TASK variable so the data from the two tasks were combined for further analysis. Because of the significance, $H1_0$ for efficiency is rejected and the alternated hypothesis accepted; dialog mode does make a difference in the time it takes to accomplish a task.

Testing of H2: The results of comparing the different treatments for the whole population are contained in Table 4.6. The results are similar to effectiveness in that the menu efficiency mean is better than the "both" but not significantly ($p>.10$). Command language efficiency mean is significantly poorer (more than a minute longer per item;

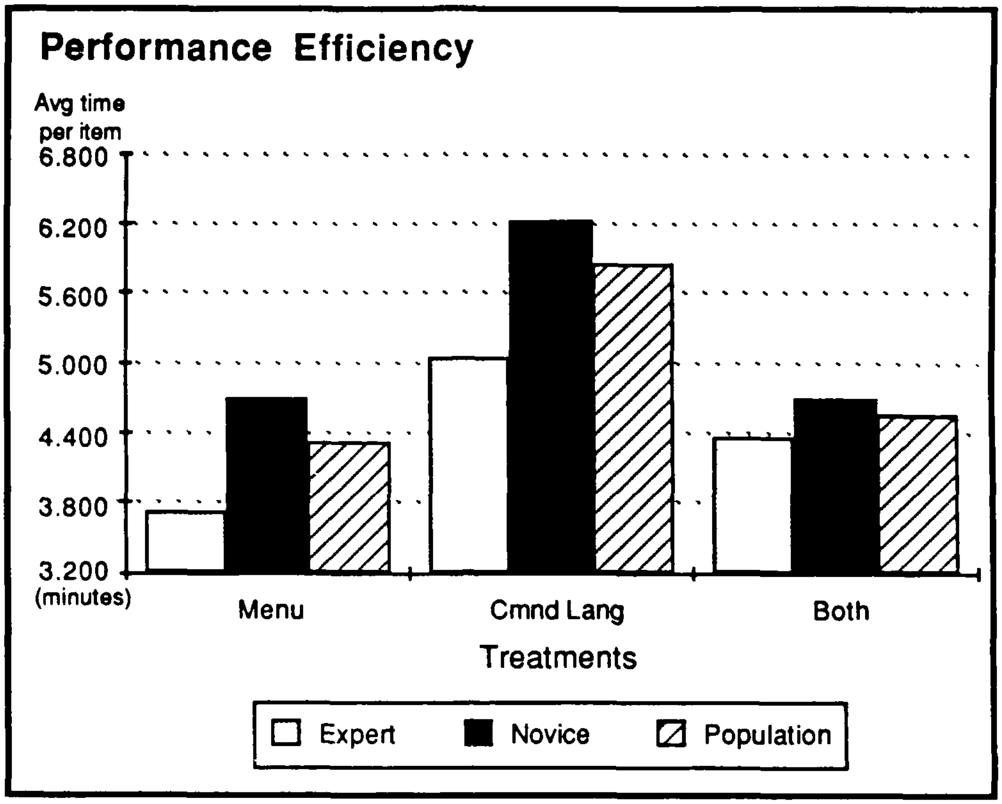


Figure 4.4

p<.001) than the other two treatments. As before, $H2_0$ cannot be rejected since "both" was not significantly better than both of the other two dialog mode treatments.

Testing of H3: In this hypothesis, experts in the "both" treatment had a mean efficiency score between the mean for those in the menu treatment (the best score) and those in the command language treatment just as in the effectiveness score (see Table 4.7). In this case, there was no significance between "both" and menu as well as "both" and command language. Since the means were all statistically equivalent and the actual menu score was better than the "both" score, the null hypothesis can not be rejected.

Performance Model - Efficiency

Source	Df	SS	F Value	p>F
Model	93	581.99	2.67	<.0001
Error	92	215.62	N/A	N/A
TECHEXP	1	29.25	12.48	<.0006
TREAT	2	63.63	13.57	<.0001
TASK	1	6.69	2.86	<.0944
TECHEXPxTREAT	2	5.67	1.21	<.3032
ID(TECHEXPxTREAT)	87	457.50	2.24	<.0001
MSE = 2.344 $R^2 = .73$				

Table 4.5

Treatment Comparisons using Total Population: Performance Efficiency

Comparisons		Means		F Value	p>F
(a)	(b)	(a)	(b)		
"Both"	Menu	4.563	4.325	1.285	>.10
"Both"	Cmnd Lang	4.563	5.852	14.904	<.001
Menu	Cmnd Lang	4.325	5.852	24.399	<.001
MSE = 2.344 F(.01) = 6.96 F(.05) = 3.96 Df 1/92					

Table 4.6

**Treatment Comparisons using Experts :
Performance Efficiency**

Comparisons		Means		F Value	p>F
(a)	(b)	(a)	(b)		
"Both"	Menu	4.370	3.744	2.088	>.10
"Both"	Cmnd Lang	4.370	5.048	2.217	>.10
Menu	Cmnd Lang	3.744	5.048	7.916	<.01
MSE = 2.344 F(.01) = 6.96 F(.05) = 3.96 Df = 1/92					

Table 4.7

**Treatment Comparisons using Novices :
Performance Efficiency**

Comparisons		Means		F Value	p>F
(a)	(b)	(a)	(b)		
"Both"	Menu	4.702	4.692	.001	>.10
"Both"	Cmnd Lang	4.702	6.236	19.458	<.001
Menu	Cmnd Lang	4.692	6.236	20.276	<.001
MSE = 2.344 F(.01) = 6.96 F(.05) = 3.96 Df = 1/92					

Table 4.8

Testing of H4: For novices, the mean for "both" and menu differ only by .01 in favor of menus for an obvious no difference (see Table 4.8). As occurred for effectiveness, the command language mean score was significantly poorer than the other two interface treatments ($p<.001$). The hypothesis is not rejected because the "both" treatment did not achieve significance over the menu treatment.

Discussion: As with effectiveness, $H2_0$ through $H4_0$ were not rejected. An interface with multiple dialog modes was not statistically better than an interface with only one dialog mode although "both" was again equal to menu in efficiency and better than command language. While better was not achieved, it is important to note that equality between "both" and menu is actually good when one considers the fact that over 1/3 of the subjects in the "both" treatment used menus less than 50% of the time.

A co-variant analysis was done with dBASE as the co-variant for the efficiency variable. The results were about the same as for effectiveness, no significance (SS=.073, F=.0311, p>.1).

The mean values for each of the dBASE classifications is shown in Figure 4.5 in a similar fashion to Figure 4.3. It clearly shows that experts in both technology and dBASE (E/E) had a much better efficiency score then all others (except for E/N in the menu treatment).

The basic results presented in Figure 4.5 show that the time to do a task increases as the subjects have less and less technical experience and

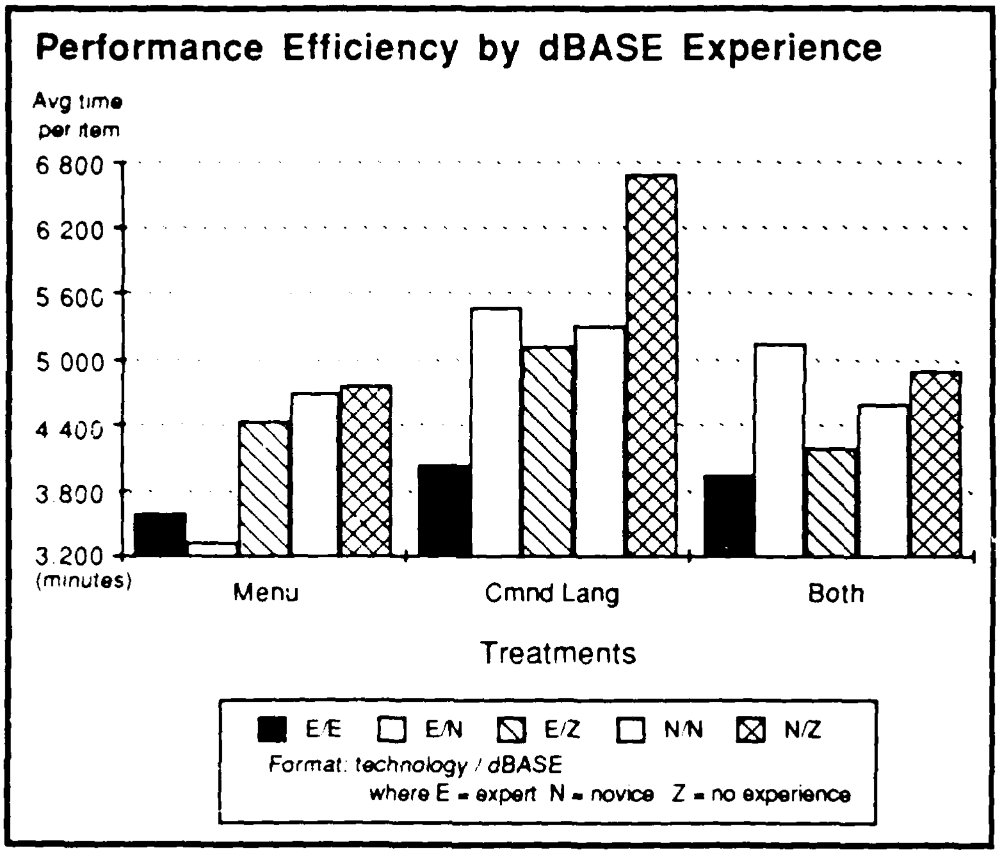
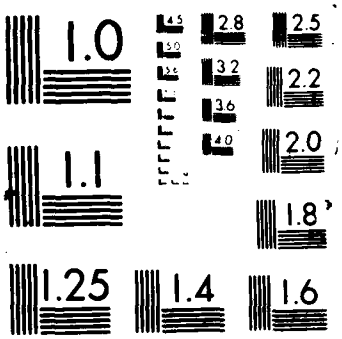


Figure 4.5

dBASE experience as one would expect. It is also interesting to see that although E/E has a much better efficiency score than E/Z in the command language treatment, these two groups had equal scores for effectiveness. That is to say, they performed equally as well but it took those with no dBASE experience longer to do the tasks (as would be expected). This shows that prior technical experience allowed the E/Z to do the correct action but it required some additional time to determine which command to use and how to use it.

For the E/N and N/N command language subjects, the poor efficiency problem seemed to be one of "a little knowledge is dangerous".

AD-A186 115 USING MULTIPLE DIALOG MODES IN A USER-SYSTEM INTERFACE 2/3
TO ACCOMODATE DIFF..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH R A TAYLOR 1986
UNCLASSIFIED AFIT/CI/NR-87-126D F/G 23/2 NL


1.0
4.5
5.0
5.6
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

Observation of the subjects during the test and subsequent evaluation of their logs showed they tended to frequently make basic mistakes with the command language syntax, which required a substantial amount of extra typing to correct the errors (a time consuming activity), and, they seemed unwilling to use the reference materials. On the other hand, E/Z subjects tended to use the handout regularly for command syntax causing less time consuming re-typing due to syntax errors. The N/Z subjects persevered in all of the treatments (no subject gave up) but many of them had a hard time with the command language causing long thinking delays and make many syntax errors. Overall discussion of the subjects in the "both" treatment will be considered in a later section.

In summary, for all the treatments, experts were generally more efficient than novices except for E/N as noted earlier. Both efficiency and effectiveness were highly correlated within treatments. The only real difference in efficiency, besides command language being much worse than the other two, was that the "both" efficiency for experts was not as good as for menu while for novice it was basically equal.

Satisfaction Hypotheses

The expected resuls from this experiment was that the subject population would be more satisfied with the user-system interface that had multiple dialog modes. This was based on the theory that the experts and novices would be able to use the dialog mode that matches their problem solving method and not be mismatched thus causing dissatisfaction. The specific hypotheses developed to test for this result are listed below. The format for H5 - H8 is very similar to H1 - H4 developed to test performance. H5 is used to see if there is an effect on satisfaction when a user-system interface has multiple dialog modes. H6 - H8 test whether or not multiple dialog modes cause greater satisfaction for different segments of the population.

Hypothesis 5

$H5_0$: An interface's dialog modes (computer directed, user directed or both) has no effect on a use·· population's satisfaction with the interface.

$H5_A$: An interface's dialog modes (computer directed, user directed or both) has an effect on a user population's satisfaction with the interface.

Hypothesis 6

$H6_0$: Subjects who have multiple dialog modes (computer and user directed mode) were equally or less satisfied with the user-system interface than users who only have either a computer directed or user directed dialog mode.

$H6_A$: Subjects who have multiple dialog modes are more satisfied with the user-system interface than users who only have either a computer directed or user directed dialog mode.

Hypothesis 7

$H7_0$: Experts who have multiple dialog modes (computer and user directed mode) were equally or less satisfied with the user-system interface than experts who only have either a computer directed or user directed dialog mode.

$H7_A$: Experts who have multiple dialog modes are more satisfied with the user-system interface than experts who only have either a computer directed or user directed dialog mode.

Hypothesis 8

$H8_0$: Novices who have multiple dialog modes (computer and user directed mode) were equally or less satisfied with the user-system interface than novices who only have either a computer directed or user directed dialog mode.

$H8_A$: Novices who have multiple dialog modes are more satisfied with the user-system interface than experts who only have either a computer directed or user directed dialog mode.

Results for Satisfaction

In all of the charts and figures presented in this section, the lower the mean score for a population the more satisfied the population is with a dialog mode interface. The scale is based on a seven point differential with 1 = extremely satisfied, 4 = neutral and 7 = extremely dissatisfied. As described in the previous chapter, the satisfaction score is the average of seven pairs of contrasting words based on the 7 point scale. The pairs are listed below for convenience (the actual format of the survey instrument that collected the data can be seen in Appendix E):

1. easy-to use	hard-to-use
2. frustrating	comfortable
3. simple	complicated

4. hard-to-learn	easy-to-learn
5. confusing	obvious
6. satisfying	dissatisfying

The last pair of contrasting words were in a question form that asked for their satisfaction level with the interface mode used in the task:

7. satisfactory	unsatisfactory

The mean scores obtained for satisfaction are shown in Figure 4.6. The data is presented so that bars that go up from the neutra response (a mean of 4) represent increasing satisfaction while bars that go down represent decreasing satisfaction (dissatisfaction). The data clearly show a difference in satisfaction levels between the treatments. In general, the command language satisfaction is in the dissatisfaction direction while the other modes are in the satisfaction direction.

Testing of H5: The ANOVA model used to determine significance is slightly different than the performance model since each subject only received one survey at the end of the experiment and not at the end of each task. Therefore, the TASK and ID(TECHEXPxTREAT) terms were dropped from the model. The results of the analysis (see Table 4.9) show that the only significance ($p < .0001$) was on treatment. The null hypothesis can then be rejected in favor of the alternate; dialog mode did make a difference in satisfaction.

Testing of H6: A comparison test of the population means (see Table 4.10) shows a very significant difference in the satisfaction level between command language and the other two ($p < .001$) even though the population satisfaction mean for command language, 4.169, is just barely

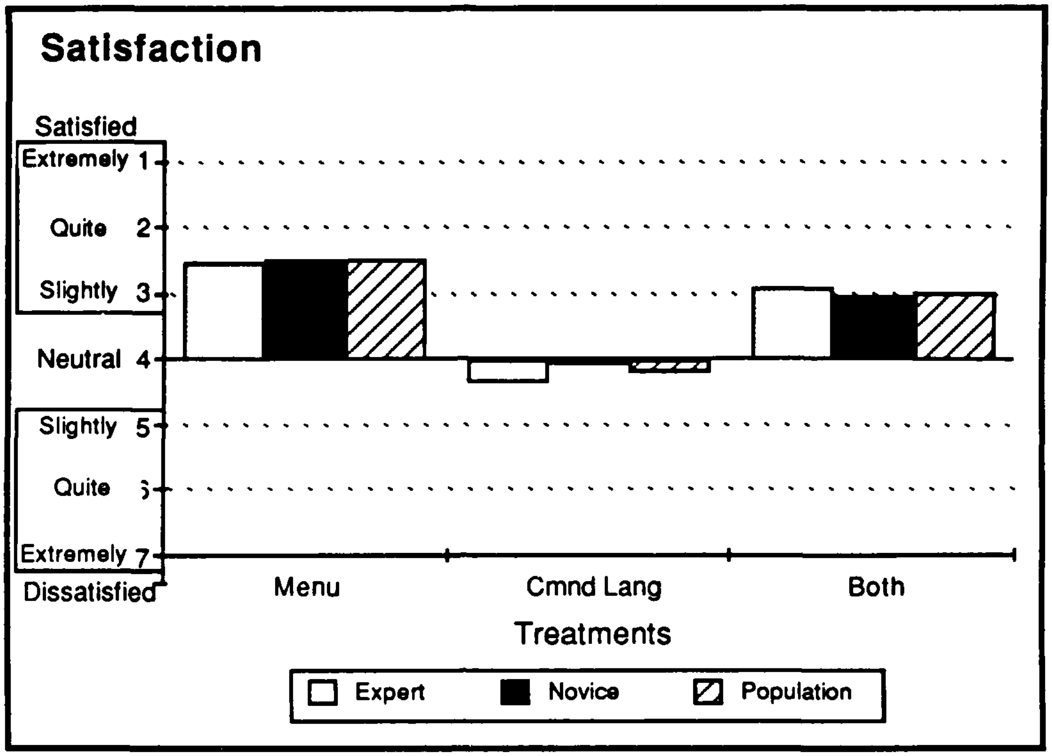


Figure 4.6

in the dissatisfaction direction (only .169). Subjects in the other two treatments indicated satisfaction with the interface from 'slightly satisfied' for "both" to almost 'quite satisfied' for menu. Since subjects were more satisfied with menus, the null hypothesis cannot be rejected. A comparison test to see if the difference was significant in favor of menu resulted in a weak significance at only the .1 level (p < .10).

Testing of H7: The means for experts, as depicted in Figure 4.6 shows that experts were more satisfied with menus than either the command language dialog mode or the interface with both types of dialog modes. This allows for the acceptance of the null hypothesis without any further analysis. Experts were not more satisfied with multiple dialog modes than with just one dialog mode, menu. The analysis in Table 4.11

Dialog Mode Satisfaction Model

Source	Df	SS	F Value	p>F
Model	5	50.44	8.64	<.0001
Error	92	107.40	N/A	N/A
TECHEXP	1	0.08	0.07	<.7875
TREAT	2	48.05	20.58	<.0001
TECHEXPxTREAT	2	0.57	0.24	<.7833
MSE = 1.167 R^2 = .32				

Table 4.9

Treatment Comparisons using Total Population: Dialog Mode Satisfaction

Comparisons		Means		F Value	p>F
(a)	(b)	(a)	(b)		
"Both"	Menu	2.960	2.465	3.11	<.10
"Both"	Cmnd Lang	2.960	4.169	20.69	<.001
Menu	Cmnd Lang	2.465	4.169	38.92	<.001
MSE = 1.167 F(.01) = 6.96 F(.05) = 3.96 Df = 1/92					

Table 4.10

Treatment Comparisons using Experts : Dialog Mode Satisfaction

Comparisons (a)	(b)	Means (a)	(b)	F Value	p>F
"Both"	Menu	2.881	2.506	0.810	>.10.
"Both"	Cmnd Lang	2.881	4.333	11.131	<.005
Menu	Cmnd Lang	2.506	4.333	17.036	<.001
MSE = 1.167 F(.01) = 6.96 F(.05) = 3.96 Df = 1/92					

Table 4.11

Treatment Comparisons using Novices : Dialog Mode Satisfaction

Comparisons (a)	(b)	Means (a)	(b)	F Value	p>F
"Both"	Menu	3.018	2.438	2.808	<.10
"Both"	Cmnd Lang	3.018	4.083	9.706	<.005
Menu	Cmnd Lang	2.438	4.083	23.770	<.001
MSE = 1.167 F(.01) = 6.96 F(.05) = 3.96 Df = 1/92					

Table 4.12

shows that while menus were more satisfying than "both", there was no statistical difference between them. It is also surprising to see that experts on the whole were basically neutral (just barely not satisfied) with the command language interface. This is not the expected result based on the literature. More will be said about this in the discussion section.

Testing of H8: It was predicted that even though menus are what the theory and literature says are preferred by novices, the interface with both menus and command language would still be more satisfying since the novice was provided a choice and not forced to use either mode. This turned out not to be the case (see Table 4.12). For novices, the menu dialog mode was more satisfying than the other two although weakly for "both" ($p < .10$); thus, the analysis requires the acceptance of the null hypothesis. In the case of novices, as with experts, "both" was significantly more satisfying than command language.

Discussion: None of the groups tested, experts, novices or the combined population, statistically showed the "both" method to be better than either of the other two although the subjects in the "both" treatment indicated satisfaction with the mode. It also showed that within treatment, there was very little difference between the expert's and the novice's satisfaction. Novices held true to the literature in regards to menu versus command language but experts did not. The experts were also more satisfied with menus than command language.

To see if this result was true for all classes of experts and novices, the satisfaction scores were graphed by dBASE experience and treatment

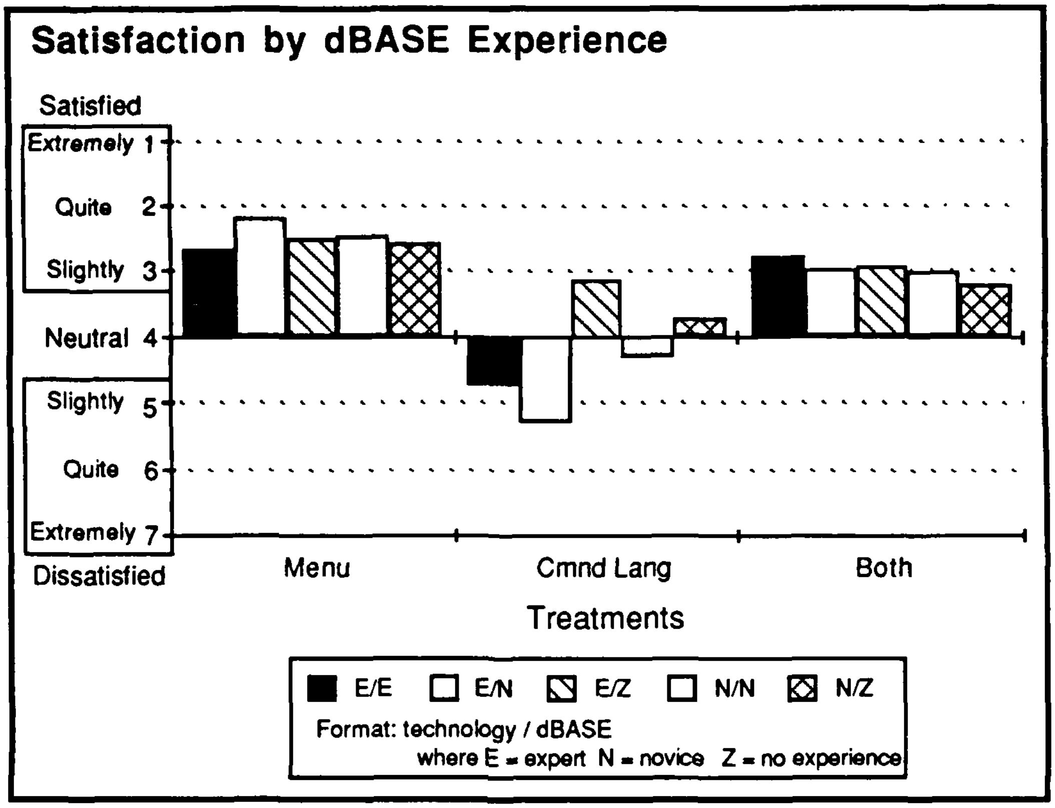


Figure 4.7

as shown in Figure 4.7 and a co-variant analysis of satisfaction using dBASE experience was done. The co-variant analysis (SS = 2.234, F = 1.914, p > .10) had no significance. The graph of satisfaction by dBASE experience showed that for menu and "both", all classes of dBASE experienced users were indeed in concert.

For command language an unusual phenomenon occurred. Those experts and novices who had never used dBASE before were actually satisfied with the interface while those who had used dBASE before were not, especially the E/N group. A review of the comments written in the post survey by subjects in the three dissatisfied groups reveled that they were bothered not only by the syntax but also by not knowing when to use which command. In other words, they were having a problem remembering the language. It was observed during the experiment that the E/N and E/E subjects tended to rely on trying to 'remember' the syntax and command use rather than take the time to use the hand-outs. On the other hand, the E/Z and N/Z subjects, who only had the brief introduction to the commands, used the handouts regularly (presumably because they did not know the syntax).

There was also a definite difference of opinion on the goodness of the handout for the command language treatment; some liked it and others did not (no comments were made about the menu handout). For example, one of the E/Z subjects noted "Handout very helpful for assigned task. It [the handout] had zoomed in on the necessary command and had good examples". Alternatively, two of the E/E subjects both indicated that the documentation was "poor" and "very poor" (the command language handout, see Appendix A, was generated from the dBASE III command reference manual with additional material written). To see how diverse the subjects in each group were on their satisfaction rating, a check of the standard deviation showed that it is only .5 for the E/E and E/N groups (means of 4.75 and 5.27 respectfully) meaning they were universally dissatisfied. The other 3 command language groups on the other hand, had a diversity of feelings within the group about command language. For N/N the standard deviation is 1.5 with a range of 1.6 (between quite and extremely satisfied) to 7.0 (extremely dissatisfied). The range for E/Z and N/Z are 1.1 to 4.7 and 2.9 to 4.9 respectfully and similar standard deviations to N/N. Subjects in the "both" treatment also had wide ranges of satisfaction within its subgroups even though the group means were at a satisfied level. The minimum value for the "both" subgroups were about the same as menu (1.1 to 1.6) but the maximum values (dissatisfaction) were 4.5 to 5.3. For menus, all of the subject subgroups were universally satisfied (maximum value was 3.8; slightly more satisfied than neutral.

One underlying assumption of the satisfaction measure in this experiment is that is somewhat of a surrogate for preference. That is, if subjects prefer menus to command language then their satisfaction should be higher in the menu and "both" treatment where they can choose menus than those subjects in the command language treatment that did not have access to the menus. To investigate this, subjects were asked in the post survey to state their preference between using a menu or a command language dialog mode. Figure 4.8 shows the results with 1 = strongly prefer menu, 4 = neutral, and 7 = strongly prefer command language. In comparing satisfaction (see Figure 4.7) to preference, all of the groups in the menu treatment were satisfied with menus and all groups prefered menus except E/N. Note that E/E and E/Z type of experts in the menu treatment prefered menus for a dialog mode interface which is contrary to the literature. For the "both" treatment, every group except E/E prefered menus and were able to choose them if desired. Command language had both E/E and E/N dissatisfied with the command language but they also prefered to use a command language for the dialog mode interface. Thus, while they were unhappy with the dBASE language,

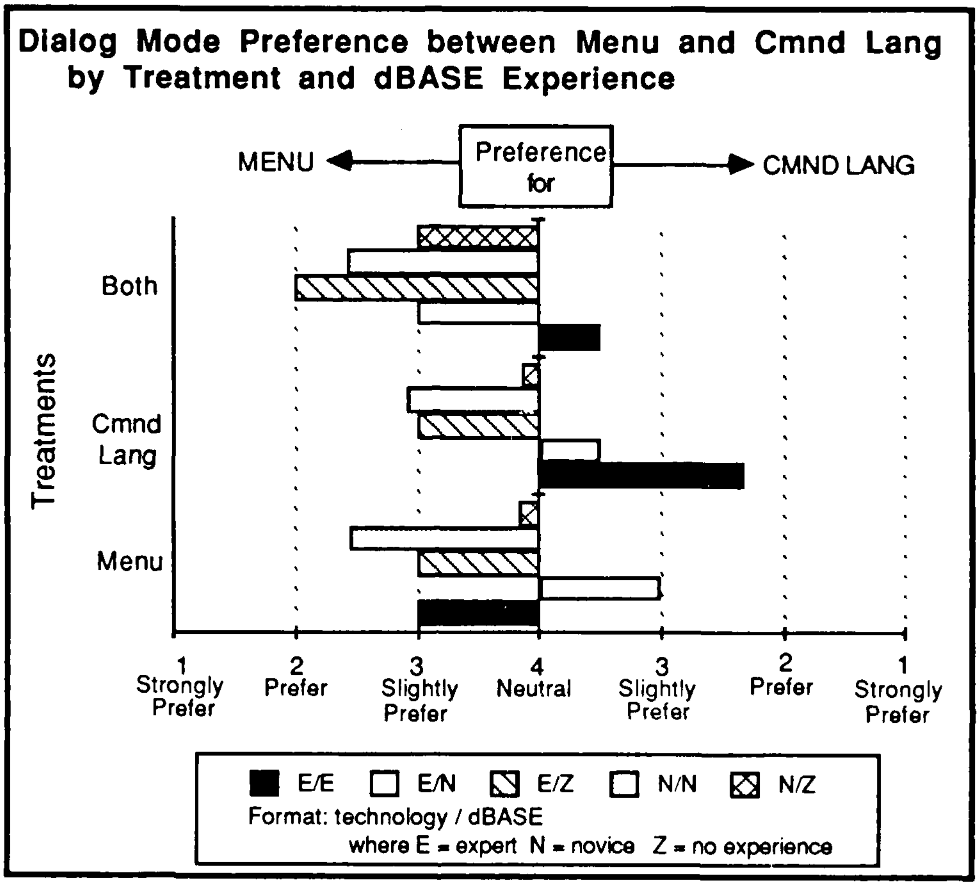


Figure 4.8

they still wanted a command language (presumable with some other characteristics than were available in dBASE III).

An interesting point to note is that the E/N subjects in the menu treatment (who had the best performance scores of any group and were the most satisfied of any group) prefered the command language while the E/E prefered menu. All of the novice groups plus the experts that had never used dBASE before prefered menu to command language.

In summary, subject satisfaction with dialog mode used in the interface did not necessarily follow what was predicted based on the literature and theory. It was also found that satisfaction and dissatisfaction with a dialog mode does not imply preference.

Application Specific Hypotheses

These hypotheses were developed to see if users, when given the choice of computer directed and user directed dialog modes, would tend to use the dialog mode that fit with their experience. That is, the experts would tend to use the user directed dialog mode and the novices the computer directed mode. The hypotheses stated that not only would each group tend toward their respective dialog mode, as the literature and theory indicate, but that they would also be more satisfied since they would be able to chose the dialog mode that fit their experience level.

Thus, there needs to be a point at which a user would transition from a novice status to an expert status and that would be the point at which the user would switch from using the computer directed mode to the user directed mode. From research done by Gilfoil [84], this transition point seemed to be around 18 hours of use in a system that was word processing and data base activities similar to this experiment. Using this research as a guideline, the following hypotheses were developed.

Hypothesis 9

$H9_0$: Subjects with more that 18 hours of application specific experience will be equally or more satisfied with only a computer directed dialog mode (menu) than those who have both or only a user directed.

$H9_A$: Subjects with more that 18 hours of application specific experience will be less satisfied with only a computer directed dialog mode (menu) than those who have both or only a user directed.

Hypothesis 10

$H10_0$: Subjects with more than 18 hours of application specific experience will not use the user directed mode more than the computer directed dialog mode.

$H10_A$: Subje‹ts with more than 18 hours of application specific experience will use the user directed mode more than the computer directed dialog mode.

Results for Applications Specific Hypotheses

One of the problems with testing these hypotheses was the lack of expert users who were also experts with the dBASE command language and, once identified, converting their specific experience to a specific number of hours of dBASE experience. The data was just not collectible on an hourly basis. Therefore, only subjects that were identified as dBASE experts by the subject classification survey were used in the analysis. It was felt that these subjects needed considerably more than 18 hours of use to qualify for this rating. Subjects that qualified as a novice in dBASE were not used since they could easily be above or below the 18 hour mark based on how they used dBASE previously.

Testing of H9: From the results obtained, it was apparent that experts in dBASE and technology (E/E) were not more satisfied with the user directed dialog mode over the computer directed menu dialog mode (see Figure 4.7). In fact, they were actually dissatisfied with the command language. Part of this dissatisfaction, but not all of it, can be attributed to being "rusty" (they had fallen back to a casual user status) with the command language. On the other hand, several of the expert subjects used dBASE on a daily basis and they were not more satisfied than the menu treatment experts. The null hypothesis cannot be rejected.

Testing of H10: The basis of this hypothesis was that since the user directed dialog mode is what the theory and literature say the expert would prefer to use, then, when given the choice, experts (especially one experienced in the command language) would select the user directed mode for performing the tasks. This is not what happened in this experiment. The E/E subjects split 50/50 on their choice and use of the dialog modes. Because of this even split, the null hypothesis cannot be rejected in favor of the alternate that experts would tend to use the user directed dialog mode.

Discussion: The theory and literature say that experts would prefer a user directed (basically command languages) over a computer directed dialog mode. To investigate this as part of these last two hypotheses, subjects preference between using a menu or a command language dialog mode was again analyzed (see Figure 4.8). This data shows that as a group, all of the E/E subjects actually do prefer (mean = 4.13) a command language when specifically asked that question although when viewed by treatment, as in the figure, one can see that that those E/E that used menus stated a preference for menus. The one problem with this data, besides a small number of subjects, is that the dialog mode that was used during the experiment might have positively biased the user to prefer that mode, especially in the menu and command language treatments. It would have been much better to have also asked this question prior to using the interface to see if a bias did develop. Since the E/E could have been biased in the first two treatments, that leaves the "both" treatment as a valid reflection of their preference since the subjects could use either type of interface and switch back and forth. In the "both" treatment, command language was slightly prefered even when half of the E/E subjects in this group used menus.

Besides the small number of subjects, there was a second problem that affected these two hypotheses. This second problem concerned several of the specific subjects used in the experiment. One of the E/E subjects that used dBASE command language on a daily basis told the monitor at the experiment that she was going to use the menu system just to see how "good" it was at facilitating the actions required to do the tasks. This subject used menus for the entire experiment even though her preference was slightly in favor of command language. Her satisfaction with the menu mode was 4.25; barely dissatisfied. Another E/E subject used the menus exclusively but also stated on the post survey "I used the menu system because it was so easy. However, I was getting bored and frustrated with it, and given another task, I would have probably tried the command mode". This subject also slightly prefered command languages over menus. Both subject were extremely in favor of having multiple dialog modes in the user-system interface. The problem exemplified by these two subjects (two of the three who used menus in the "both" treatment) makes the results somewhat suspect and clearly points the need for a larger subject population for investigating these two hypothesis.

Regardless of these two problems, if all of the subjects that have been exposed to dBASE command language are looked at in the "both" treatment, the menu to command language usage was 14 to 10 or a 60% to 40% usage in favor of menus. Further analysis of the subjects in the "both" treatment is presented in the next section.

"both" Treatment

The subjects in the "both" treatment were analyzed further by how much of each dialog mode they used in accomplishing the tasks. To determine this, the subjects in the "both" treatment were broken out by how much of each dialog mode they used to perform the tasks. This was done in two ways which turned out to be basically equivalent.

First, the total number of carriage returns entered using the command language were counted and recorded as a percentage of total carriage returns entered. The second way was to determine what percent of the task was done using command language by looking at how each item in the tasks was completed. Only completed items were included in the scoring. On the average, the first method underscored the second by about 15% but by just counting carriage returns the command language was penalized. One carriage return could accomplish the same action that required a minimum of three to four carriage returns with menus thus giving a higher percentage to menu use. Because the results were relatively the same and because of the bias against command language using carriage returns, the second method was used for analysis.

The subjects in the "both" treatment were divided into expert and novice and then again subdivided into those who used the command language less than 25% of the time to do the tasks (e.g. used menus more that 75% of the time) and those who used command language more than 47% of the time (there were no subjects between 25% and 47%). The latter group is labeled "both"/command language (BC) and the former "both"/menu (BM). Once this separation was done, each classification of groups of subjects in the "both" treatment, from the total population down to technology/dBASE experience grouping, were looked at to see what percentage of each group fell into the BM and BC category (see

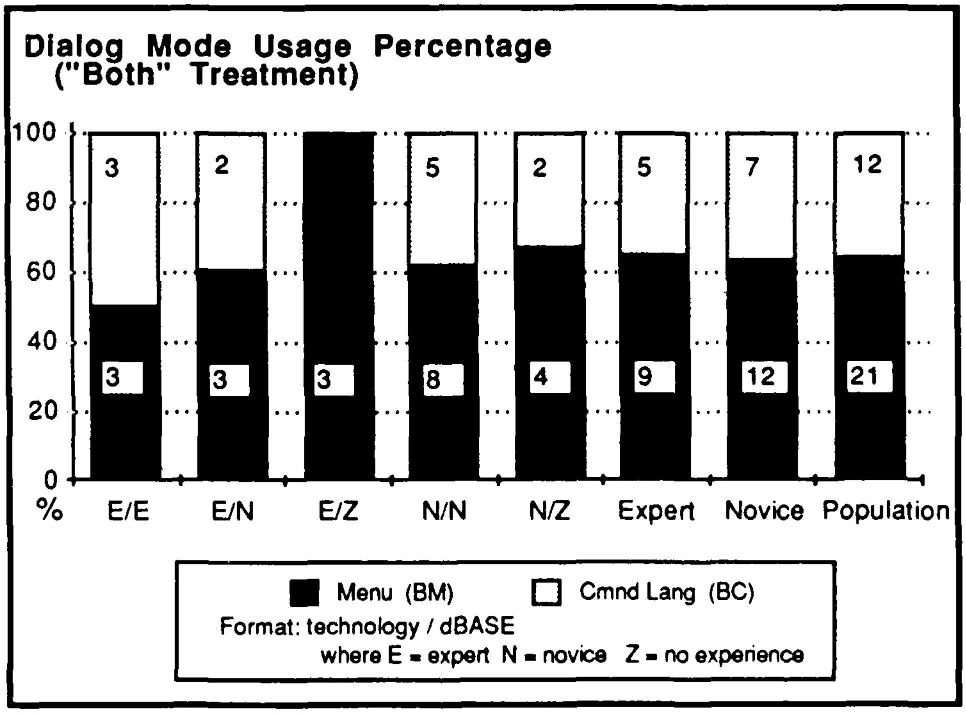


Figure 4.9

Figure 4.9). From this analysis, one sees that the groups were all basically around a 60 - 40 split of BM to BC respectfully except for E/Z all of whom used the menus exclusively. Of the total population of 21 subjects that used menus more than 75% of the time, half of them used menus exclusively. For the BC population, 3 of the 12 used the command language exclusively. The point of this data is that the 33 subjects that had the user-system interface with both menus and command language had a dialog mode usage that ranged from 100% menu to 100% command language with all sorts of combinations between. What must be carefully pointed out though is that the command lɛnguage users were not exclusively the dBASE or technology experienced subject. The choice of what mode to use did not seem to be based on expert novice differences as the theory and literature suggested. The choice seemed to be more of a preference based on not only experience, but ease of using the dialog mode, the specific task to be done, curiosity (one E/E subject said that she initially used menus just to see how good it was and not necessarily because of preference or satisfaction requirements.), the mental effort required to do the task, learning and especially remembering (memory and recall).

From the performance data in Table 4.13 it is clear that having a choice to use both dialog modes had an effect on satisfaction and performance both positively and negatively. There was a small improvement on the efficiency scores of the two "both" subgroups (BM and BC) but at the same time there was a reduction, though only slightly, on the effectiveness scores for the BC population. For satisfaction, the

Comparisons of Dialog Mode Usage in the "both" Treatment Population to the other Populations

Population	Effectiveness	Efficiency	Satisfaction
BC - "both"/ cmnd lang	2.64	5.37*	3.74*
CC - cmnd lang treatment	2.73*	5.85	4.17
Percent difference	0.03%	0.08%	0.10%
BM - "both"/ menu	3.22*	4.05*	2.51
MM - menu treatment	3.07	4.32	2.46*
Percent difference	0.05%	0.06%	0.02%

* = best score

BM = experts and novices who used menus more than 75% of the time

BC = experts and novices who used menus less than 50% of the time

Table 4.13

BC population was barely satisfied while the command language treatment (CC) population were barely dissatisfied. In the menu treatment (MM) and BM subgroups, the difference was only an insignificant .05 but both scores reflected high satisfaction with the dialog mode.

In analyzing the keystroke logs of those subjects that used the command language more than 47% of the time, it was found that generally they only switched to the menu system when they could not figure out how to correctly accomplish the item using the command language. Written comments by subjects support this observation. Some examples of why subjects switched were provided by subjects on the post surveys. An E/E subject stated "I switched frequently. I know query commands in dBASE well, but some of the queries requested were very difficult. For these, I would try to use command language, would bomb and would then switch to menu mode. I liked being able to switch. It gave me multiple approaches to solving the problem." An N/N subject said "I switched back and forth since some tasks were easier with menus and others with command language."

When novice performance scores alone are looked at, the differences between those who selected the menu system and those who had to use menus is significant (p<.02). Novices who chose to use menus outperformed those who had to use menus. Those who chose command language performed equally as well as those who had to use command language such that overall, "both" was better for novices. From this and written comments, it seems that these subjects that chose menus felt more comfortable then some of those forced to used menus and thus performed better. One of the subjects forced to use menus (a novice with no dBASE experience) stated on the post survey that he felt uncomfortable with menus and felt that he would have done much better on the tasks if he could have used the command language.

A more detailed look at satisfaction shows in Figure 4.10 that the satisfaction for menu users in the menu treatment and the "both" treatment had comparable results with no significant differences. For command language, Figure 4.11, there was a large difference in expert satisfaction level with those who used the command language in the "both" treatment being much more satisfied than those who used it in the command language treatment. In Figure 4.11, there were no E/Z subjects in the BC population thus no score is listed. The BC value for

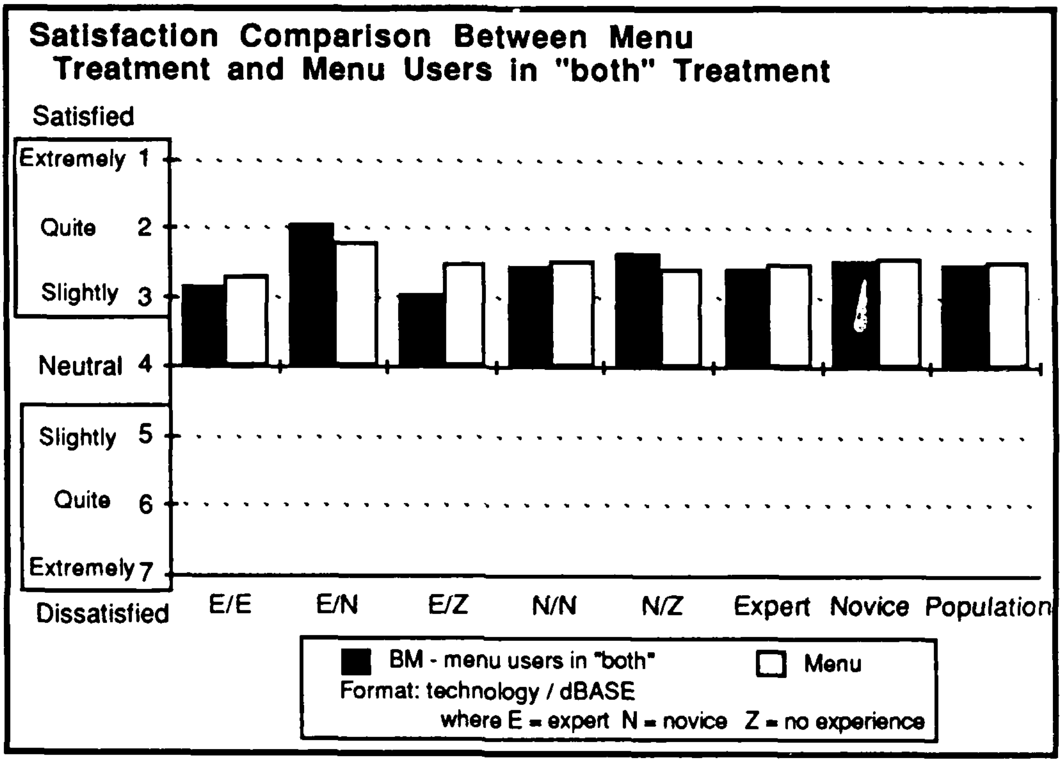

Satisfaction Comparison Between Menu Treatment and Menu Users in "both" Treatment
Satisfied
Extremely 1
Quite 2
Slightly 3
Neutral 4
Slightly 5
Quite 6
Extremely 7
Dissatisfied
E/E
E/N
E/Z
N/N
N/Z
Expert
Novice
Population
■ BM - menu users in "both"
□ Menu
Format: technology / dBASE
where E = expert N = novice Z = no experience

Figure 4.10

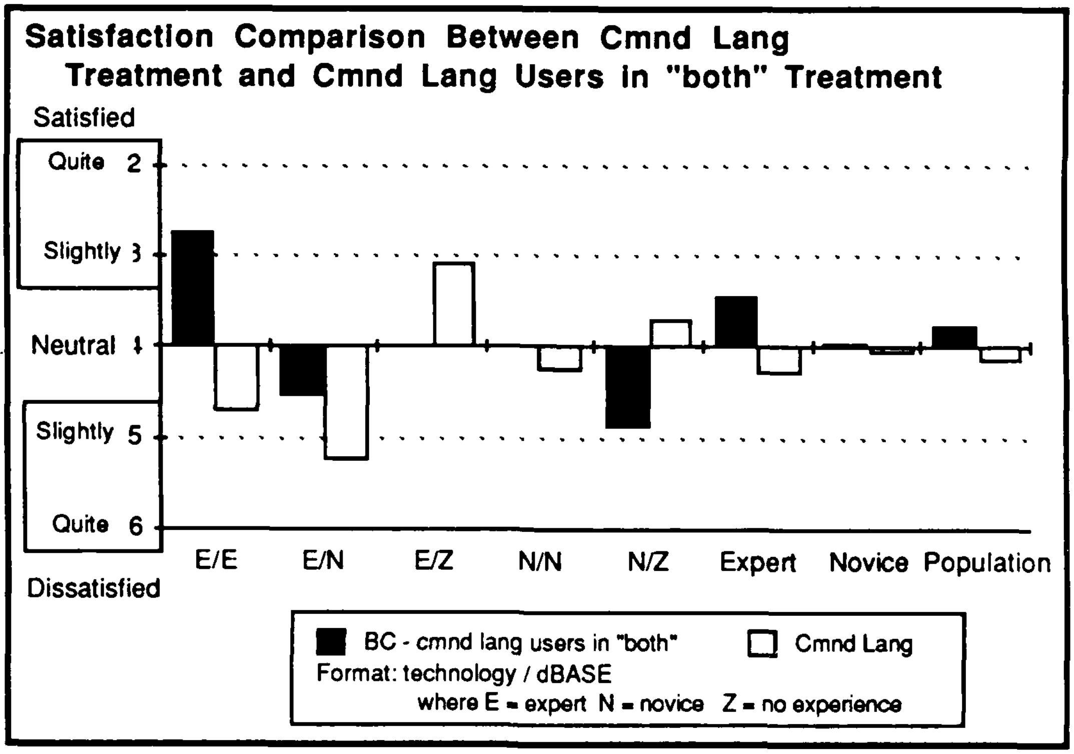

Satisfaction Comparison Between Cmnd Lang Treatment and Cmnd Lang Users in "both" Treatment
Satisfied
Quite 2
Slightly 3
Neutral 4
Slightly 5
Quite 6
Dissatisfied
E/E
E/N
E/Z
N/N
N/Z
Expert
Novice
Population
BC - cmnd lang users in "both"
Cmnd Lang
Format: technology / dBASE
where E = expert N = novice Z = no experience

Figure 4.11

N/N is 3.98 and for novices 3.95. Thus for command language, satisfaction was improved in all cases ,except N/Z, when the subject chose command language rather than being forced to use it.

Summary of Hypothesis Testing

The experimental results showed that the dialog mode in a user-system interface can make a difference in a user's performance for accomplishing a task and their satisfaction with the interface; thus, H1 and H5 were rejected and the alternate hypotheses accepted. For H2 - H4 (performance) and H6 - H8 (satisfaction), the alternate hypotheses were strongly stated in that multiple dialog modes in a user-system interface were tested for being better then either of the single dialog mode interfaces instead of being just equal (a weaker statement). The analysis of the data showed, in all cases, that multiple dialog modes were not statistically better then both of the other cases; therefore, none of the null hypotheses were rejected in favor of the alternate. If the hypotheses had been stated in a weaker form, i.e. multiple dialog modes are equal or better than one mode, then many of the null hypotheses would have been rejected for the alternate.

Testing of the applications specific hypotheses, H9 and H10, showed that experts did not favor command language over menus in satisfaction or for accomplishing the tasks. Because of this, the null hypothesis could not be rejected for either hypothesis. These particular results are not as strong as those obtained for H1 - H8 because of the small sample size.

CHAPTER V

Conclusions and Implications

Conclusions

The hypotheses for this experiment were based on a normative theory developed from the literature and past experiments. This normative theory said that computer directed user-system interfaces (e.g. menus) were best for novices and that they would be satisfied and perform well with this type of interface. The theory also said that experienced users would not be satisfied with a computer directed interface because they would want to be more in control and thus would do better if they had a user-directed interface such as command language. The underlying basis for this normative theory was conjectured to be grounded in the theory developed around the differences in expert and novice problem solving strategies. These strategies seemed to be closely aligned to user directed and computer directed styles of dialog modes. Expert versus novice problems solving strategy was in turn based on the basic issues of human memory and learning. From all of this information, it was hypothesized that the reason a user-system interface with a user directed dialog mode seemed best for experts was because it lent itself to the expert's problem solving strategy. The same is true for novices with a computer directed dialog mode. Thus, if there was a mixed population of expert and novice users, then a user-system interface with both a user and a computer directed dialog mode would provide more satisfaction and facilitate better performance than just one dialog mode that all users would be forced to use.

In this experimental task setting of data base update, query, and report generation, the results of this experiment did not support the normative theory or the expert versus novice problem solving theoretical foundation; thus, most of the hypotheses were rejected. Specifically, experts were not dissatisfied with a menu user-system interface and performed quite well with menus. Many experts were in fact dissatisfied with the command language, even those with extensive dBASE experience, while some novices were not dissatisfied. Most significantly, when the experts and novices were given the opportunity to choose their dialog mode, as in the "both" treatment, 60% chose menus with the remaining 40% choosing command language in both the expert and novice populations. What this experiment indicates is that there are other reasons for a user's choice and satisfaction in using a particular type of interface than those presented in prior literature and that expert versus novice problem solving strategies may not be the underlying basis for this choice.

The next two sections of this chapter analyze how the results of this experiment affect the normative theory developed from the literature and the theoretical foundation of expert versus novice problem solving strategies. The last sections will discuss some implications of these results for user-system interface design and future research topics for continuing this line of research.

Experimental Impacts on the Normative Theory

A key result in this experiment that casts doubt on the developed normative theory, found in a preponderance of user-system interface literature, is the universal lack of dissatisfaction among the subjects with the computer directed menu dialog mode. Experts, even those with dBASE experience, and novices were both satisfied with the menu system. Analysis of the experiment and the results indicate that there are two primary rea‿ons for the lack of experimental support for this normative theory and may suggest that the normative theory no longer applies especially for settings similar to this experiment (e.g. data base activities).

The first reason has to do with the response time of the menu system. Most of the literature and prior experiments base their requirement for a different user-system interface for experts on results and observations based on menu systems used on mainframe computer systems. These systems are typically time sharing systems where users can expect to have delays in presenting and responding to menu input because of such things as terminal line speeds and contention for service among other users. An important factor in this experiment was the speed of the dedicated micro system for each subject that displayed the full screen menus. It was exceptionally fast even though it painted the screen. The subject was also allowed to type ahead if they so desired. These features allowed subjects to rapidly progress through many of the options menus and type in data requirements while the menu was being generated. This speed of processing is one of the activities prized by many users especially ones that are experienced. Speed of processing is also one of the main advantages of command language, e.g. one does not have to progress through many levels of menus to perform an action. Thus, for experts, the speed of the menu seemed to have been sufficiently fast that they were satisfied. If this same system had been implemented on a mainframe computer where there was a definite wait time for each menu to be displayed (especially on a terminal using 1200 baud), the experts' satisfaction (and maybe even the novices') may have been much lower and narrowed the satisfaction difference between command language and menus. It might also have caused more of the experts in the "both" treatment to use the command language. Further analysis in this area is needed before any specific conclusion can be generalized beyond the micros used in this experiment but it does seem that micros don't support the theory. It may also be that experts are not necessarily dissatisfied with computer directed interfaces just dissatisfied with poor design and response times for menus. On mainframes, one way for experts to correct for slow response with menus is to use command language since no delay is generated waiting for the menu to be presented.

The second reason has to do with advances made in identifying human factors that need to be incorporated into the interface design as was done with the menu system developed for this experiment and as exists to some extent in the dBASE command language. These factors are the underlying concepts described in Chapter II of which many were developed simultaneous with literature describing the need for different user-system interfaces for experts and novices. They were not features readily found in many of the systems of the late 70's and early '80s when the norma ive ideas were being developed. By applying these design principles to the menu dialog mode, many of the problems experts had with menus in the past, such as slow response and inadequate flexibility, seem to havc been overcome. On the other hand, even though the dBASE command language had many excellent human factors features, there was still dissatisfaction. Much of the dissatisfaction seemed to be directed at not having some sort of function to assist users recall syntax (help was available but rarely used) and not having a mechanism to correct syntax errors other than retyping the entire command line.

In summary, new technology and the incorporation of good design techniques based on experimental and human factors studies seem to have overcome the need for providing experts with a user directed dialog mode. This research shows that a computer directed menu interface can satisfy both novices and experts alike.

Experimental Impacts on Expert vs Novice Problem Solving Strategies and Dialog Modes

Subjects in the "both" treatment did not select the dialog mode predicted by their problem solving strategies. Novice in technology selected both menu and command language, when given their choice, without regard to previous experience with dBASE. The same held true for experts. It was also found that even when experts used the menu user-system interface (which favors a novice mode of problem solving) they performed better than those who used the command language. The experimental results did not provide evidence to support the difference in expert and novice problem solving techniques as the basis for why experts and novices need different type of user-system interfaces.

A user activity that was not expected was the switching back and forth, by both novices and experts alike in the "both" treatment, between menus and command language depending on the task required. Of the 12 subjects in this treatment that used command language more that 47% of the time, 25% used the command language exclusively. It was felt, based on the literature and the theory, that subjects would basically use one or the other of the dialog modes. This was not the case. One of the novice subjects that was also a novice in dBASE commented about this switching on the post survey by saying: " Since I was able to use both modes I liked [the] method. I found that when I needed to find a particular record, the "command" was easier then menu. On the other hand, the actual modifying, adding tasks were easier with the menu mode!" A previously quoted expert in both dBASE and technology said he switched to menu if it was too hard a task with command language or he couldn't get the correct answer using command language. Thus, it seems that having two different dialog modes provided some of the subjects with alternative methods for accomplishing their task instead of being stuck with just one method that they didn't like or couldn't use for the required action. For those subjects that didn't switch back and forth, there was no apparent relationship between experience and the dialog mode chosen. One of the novice subjects seems to have captured the essence of the "both" user's and their switching by noting on the post survey:

> "If you work with dBASE a lot, the command mode is easier and faster, not so repetitious. However, if you are not constantly using dBASE, then it is extremely tedious and frustrating to use, i.e. it's too picky. This was a good experiment, definitely pointing out the pros and cons of each. At my level, I preferred the menu driven mode but it is nice to be able to change for some things. "

This particular comment, plus results and observations made during the experiment, provides support for the User Experience Level Transitions model (Figure 2.3), as applied to the dBASE environment on learning stages and casual users. Some of the problems encountered by dBASE experienced subjects who used the command language seemed to be caused by their having fallen back to a casual user status through lack of recent use that could have kept them at the expert level for recalling and using the commands. They did not lack general experience with dBASE or technology, they just lacked adequate current experience that caused them to have a problem remembering how to form the syntax. Overall, menus for a casual user worked well whereas command language for the casual user seemed to cause frustration.

This experiment showed that the dialog mode could make a difference in a user's performance, both effectiveness and efficiency, and in their satisfaction with the dialog mode. It also showed that the subjects' choice of which dialog mode to use was not dependent on the subjects' technology experience level and associated problem solving strategy but more on a preference for one or the other of the dialog modes or the task requirements. Experience with dBASE III had both a positive and negative influence on performance and satisfaction for all classes of user for a variety of different reasons. When asked about their preference for only one mode versus having both modes, the vast majority of the subjects expressed a strong preference for a user-system interface with both dialog modes (see Figure 5.1), even though they were most satisfied with menus and performed best with menu. Those subjects that were in the treatment that provided them this preference for multiple dialog modes and made their choice, could not exceed the scores for the subjects forced to use menus. Thus, having their preference did not improve satisfaction and performance.

Overall, the research indicates that the differences in expert and novice problem solving strategies did not seem to impact dialog mode usage and satisfaction as predicted by the theory and literature. Some reasons for this may be that the dialog mode was possibly at too low a level of thought processing to impact on the problem solving tasks required in the experiment or there just isn't a correlation between expert versus novice problem solving strategies and user-system computer interfaces. The important considerations in using a dialog mode seem to be based more on previous experience with dialog modes,

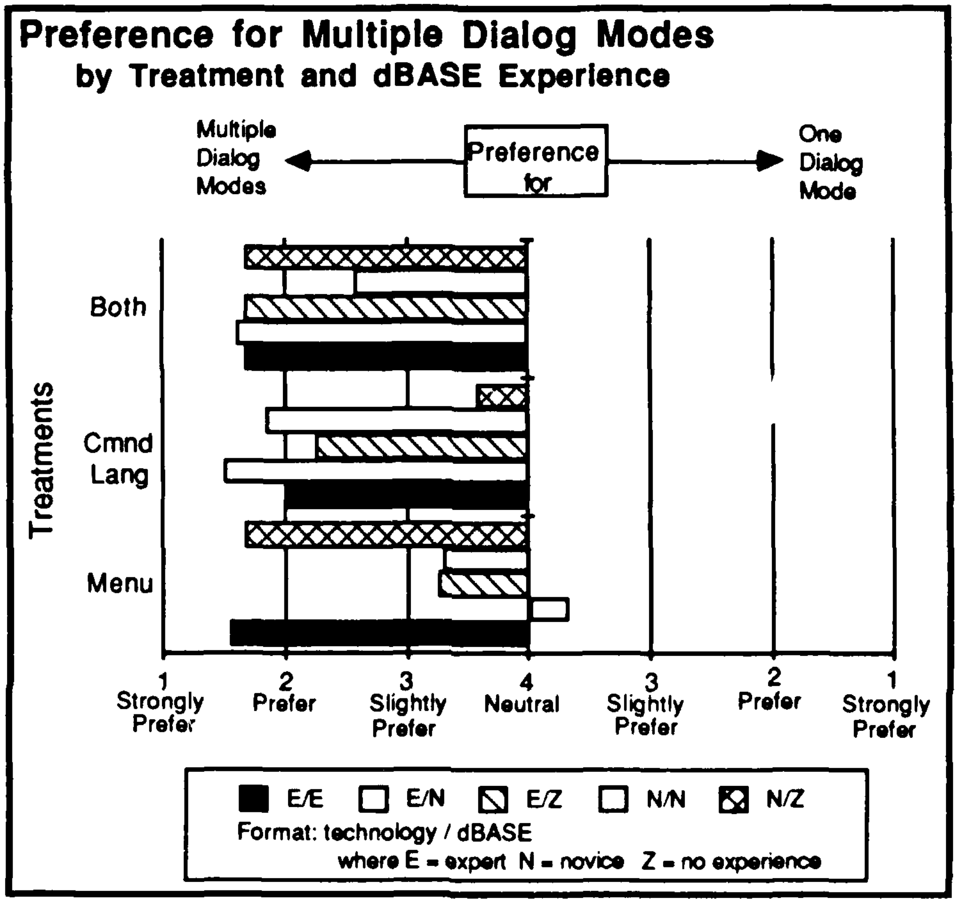


Figure 5.1

ease of using the dialog mode for the task, speed, and a bias either for or against a specific dialog mode or dialog style based on a subject's own unique set of preferences and general experiences.

To summarize the important points as succinctly as possible, the following conclusions were reached with regards to this research:

(a). The results did not provide evidence to support the difference in expert and novice problem solving techniques as the basis for why experts and novices need different type of user-system interfaces.

(b). The results did not support the normative theory developed from other researcher's opinions, observations, and research that experts would find menus unsatisfactory as a user-system interface and prefer a user directed dialog mode.

(c). A user-system interface with multiple dialog modes for a population with varried experience levels was not statistically better then an interface with menu only but was better than command language only.

(d). A user-system interface with only menus was consistently better for experts in both performance and satisfaction over the other user-system interfaces used in this research.

(e). A user-system interface with only a command language was consistantly worse for novices in both performance and satisfaction then either of the other two interfaces used in the research.

(f). Subjects preferred multiple dialog modes but, when given multiple dialog modes, generally did not perform as well as those with menus only.

(g). When subjects given multiple dialog modes, choice of mode to perform tasks was not correlated to either technology or dBASE experience.

(h). Satisfaction with a dialog mode seems to be based on previous experience with dialog modes, ease of using the dialog mode for the task, speed, and a bias either for or against a specific dialog mode or dialog style based on a subject's own unique set of preferences and general experiences and not just technology and application specific experience.

Implications

The results of this experiment provide situational type of information that can have an impact on a user population's satisfaction and performance with a user-system interface. In particular, the results clearly shows that between the dialog modes used in the research, the menu mode is an excellent choice for novices just as has been found with other experiments. Novices performed best and were most satisfied with the menus. More importantly though, it was found that experts in technology also did best with menus. This was especially true for those that were novices to the dBASE command language or had fallen back to casual user status. This indicates that for applications that have periodic data base activities and are to be used primarily by casual and novice users to the application (regardless of their technology experience), a well structured menu system that provides all of the necessary functionality and utilizes current human factors concepts will allow the user population to perform better and be more satisfied than with a command language dialog. Having only a command language can actually lead to dissatisfaction and poorer performance with the application. Thus, under these types of conditions, there is no real need for having a user directed dialog mode as well as the computer directed mode. Although, it should not be cast aside without due consideration since there is a strong preference for both modes and the performance and satisfaction were basically equal.

There are two other important items that have implications for user-system interfaces. The first is that having multiple dialog modes does have the advantage of providing the user with alternative methods for accomplishing the same activity. It allows the user to "choose" which method to use under which conditions based not on technology experience but on some other set of experiences and preferences.

The second item that can be stated is that if the user population is a mix of user experience for an application and a command language is required for what ever reason, then better performance and satisfaction will be obtained from the casual and novice users (and even some of the experts) if a menu system is also included in the user-system interface. To make this true though, the user-system developer must use design principles found in the literature and ensure that the response time of the menus is similar to that achieved with the command language.

Future Research

There are a number of experiments that can build on this research of which the most promising ones are described below. Some of the research items described below were specifically held constant for this research as they were not the primary interest for this study.

Of particular interest would be to study the reasoning behind a subject's choice in selecting a dialog mode for satisfying a task. This research would initially require each subject to perform the same task with two or more different dialog modes, varying the order to eliminate order bias. Then the subject would accomplish a final task with a user-system interface that contained two or more of the original modes. By first having the subject use each dialog mode exclusively for a task, performance and satisfaction measures could be gather after each treatment to compare to the final treatment that used multiple dialog modes. Plus, the subject would be in a better position to more accurately describe his satisfaction with having multiple dialog modes and the reasoning for why he chose as he did in the final task. The research would also show whether or not conducting an experiment in this fashion had any impact on the number of subjects that switched back and forth between dialog modes when more than one mode was available.

A second area that would provide insight to user-system interfaces would be to directly address the issue of whether or not a subject's problem solving strategy can be influenced, positively or negatively, by a dialog mode. The research would first require an analysis of the subject's problem solving strategy in a problem domain that could also be computerized (text editing for example). The subjects would then perform a similar problem in the same domain using a computer with a user-system interface that had either a computer directed dialog mode, a user directed dialog mode or both modes. Using protocol analysis of the subject's methodology to complete the computer task, the researcher can determine if: (1) the dialog mode influenced the subject's problem solving strategy, (2) the dialog mode can improve or hinder the subject's problem solving performance, and (3) there is any relationship between dialog modes and problem solving strategies

A final area of interest would be to concentrate on the dialog tasks. The dialog tasks in this research were data entry, text editing and query set in a data base activity. Additional research needs to be conducted to determine if there are certain dialog modes that cause users to be more satisfied and perform better for specific types of dialog tasks. It could be the case that some tasks are much better for menus and others for command languages. If so, then if the overall activity requires tasks that are best done by different dialog modes then multiple dialog modes or a blend of different dialog modes could be the most advantageous user-system interface. Support for this new research has already been found and commented on earlier (e.g. the subjects comments on switching modes depending on what needed to be done).

There are many other research topics that can build on this and other research besides the few mentioned here. The issues of which type of user-system interface is best for what type of user in what type of situation has not been solved but progress is being made. Use of good experimental design and techniques and the use of established frameworks like Benbasat [81] to guide the research will definitely enhance our ability to further our knowledge of this extremely important facet of human computer interaction.

Appendix A

dBASE II Command Language Handout

Database Structures

The address book maintenance database contains two databases. The address book database, called 'addbook' contains the name, address and telephone numbers of a number your friends. The other database, called 'xmaslist', contains information on who you sent Christmas cards to this past year and who sent one to you.

The name of the fields in each of the databases and their associated characteristics are described below. All fields are character data.

ADDBOOK

Name	Size	Comments
last_name	10	
first_name	10	
spouse_n	10	husband or wife first name
maiden_n	10	wife's maiden name
address	20	
city	14	
state	2	two letter abbreviation
zip	5	
area_code	3	
home_tel	8	format -> nnn-nnnn
work_tel	8	format -> nnn-nnnn

XMASLIST

Name	Size	Comments
last_name	10	same as last_name or maiden_n above
first_name	10	same as first_name or spouse_n above
sent	1	you sent this person a card (y/n)
received	1	you received a card from this person (y/n)

Classes of Commands

Creation and Manipulation of Files

The following commands create database files and associated files.

INDEX	-	Creates an index file
JOIN	-	Combines specified records and fields from two database files
SELECT	-	Used to define a database work area
SORT	-	Creates a sorted version of the active database file
USE	-	Specifies the database file to be used for all operations until another USE command is issued

Update of Database

The following commands add, delete or modify data records in database.

APPEND BLANK	-	Adds a blank record to the end of a database file
DELETE	-	Marks records for deletion from database file
PACK	-	Removes records marked for deletion
RECALL	-	Reinstates records marked for deletion
REPLACE	-	Replaces data fields in a record(s) with specified values

Query and Reports

The following commands display and list data.

DISPLAY	-	Display records and fields (pauses while displaying)
LIST	-	List records and fields (does not pause while listing)

Positioning Record Pointer

The following commands position the current record pointer to records as directed.

CONTINUE	-	Positions to the next record with conditions specified in the LOCATE command

GOTO	-	Positions directly to a specified record
LOCATE	-	Positions to a record that fits a specified condition

Information, Environment and Parameter Controls

The following commands provide database information, control the environment or set system control parameters.

CLEAR	-	Erases the CRT screen
CLOSE DATABASES	-	Closes/clears all database work spaces and aliases
DIR	-	Shows files on the current default disk d-ive
DISPLAY STRUCTURE	-	Display the structure (field names and sizes) for current database in USE
SET HEADING ON/off	-	Field names DISPLAY/do not display above the fields in LIST or DISPLAY
SET INDEX TO	-	Opens named INDEX files

Syntax of Commands

The structure of a command is its syntax. Each command begins with a verb which is the basic command. Many commands also have one or more clauses that tailor the command to meet a particular need. The general syntax of a command is illustrated by:

```
VERB[<scope>] [<expression list>] [FOR/WHILE <condition>]
```

The specific syntax for each command is shown in the Command Reference section.

The items enclosed in square brackets are optional. They may be entered in any order. Capitalized items are entered exactly as shown in the command section. Items enclosed by the angle brackets, < >, are selected by the user.

Do **not** type the square brackets or angle brackets when entering a command. Words shown in upper case indicate a reserved word and must be typed as shown. They need not be typed in upper case when used in dBASE.

The definitions of the key words used in the syntax example are:

VERB Command name for specifying a dBASE action. It is a reserved word and shown as uppercase throughout this document.

scope An optional part of many commands that specifies the amount of the database to which the command applies. Each command which has the scope option has a default of either the current database record or all database records. The scope may be:

n	- a single record
NEXT n	- n records beginning with the current record
ALL	- all of the records in the database

expression (exp) a field name, function or a constant.

expression list one or more expressions separated by commas.

FOR <condition> indicates that the command is to apply to every record in the database for which the condition is true.

WHILE <condition> indicates that the command is to be repeated as long as the stated condition is true.

condition defines more specifically what a command is to do. The condition itself is a comparison between two or more items. Only those records for which the condition is true will the verb be applied. Multiple conditions may be defined by using the logical operators .AND. and .OR.

COMMAND LANGUAGE RESTRICTIONS

1. Only use commands listed in the handout.

2. APPEND, EDIT, and BROUSE command are not to be used to add, change or delete records. Use one of the following methods instead:

a. Add a record

```
APPEND BLANK
REPLACE  fieldname WITH 'data', [fieldname WITH 'data',...]
   e.g. REPLACE   last_name  WITH  'TAYLOR'
```

b. Modify one or more record

```
LOCATE  FOR  criteria
REPLACE ______
  [CONTINUE
   REPLACE ____ ]
```

or

```
REPLACE  fieldname WITH  'data'  FOR  criteria
```

c. Delete one or more records

```
LOCATE   FOR criteria
DELETE
  [CONTINUE
   DELETE ]
PACK
```

or

```
DELETE   FOR  criteria
PACK
```

3. For reports, use DISPLAY and LIST commands.

4. For sorted reports, use SORT or build an INDEX.

5. Execute the following command after you are finished with each numbered item in the experimental task:

```
DO  ENDITEM
```

(Note: The rest of the hand-out contained documentation on each of the commands listed in the previous paragraphs. Only a sample set of the commands are included to show the style and format of the documentation)

OPERATORS

dBASE III provides four types of operators: mathematical, relational, logical, and string.

MATHEMATICAL OPERATORS Mathematical operators generate numeric results.

+	= Addition.
−	= Subtraction.
•	= Multiplicatior.
	= Division.
•• or ^	= Exponentiation.
( )	= Parentheses for grouping.

RELATIONAL OPERATORS Relational operators generate logical results (for example, True or False). Relational operators can be used with either character, numeric, or date expressions. Both expressions must be of the same type.

⟨	— Less than.
⟩	— Greater than.
=	— Equal.
⟨ ⟩ or #	— Not equal.
⟨=	— *Less than or equal.*
⟩=	— Greater than or equal.
$	— Substring comparison. (For example, if A and B are character strings, A$B returns a logical True if A is either identical to B or contained within B.)

LOGICAL OPERATORS Logical operators obtain a logical result from comparing two logical expressions.

.AND.	— Logical and.
.OR.	— Logical or.
.NOT.	— Logical not (works with a single expression).
( )	— Parentheses for grouping.

STRING OPERATORS

+	— Concatenation operator. It is used to join two or more character strings into a single character string.
−	— Concatenation operator. It is used to join two or more character strings into a single character string. All trailing spaces are moved to the end of the combined string.

DELETE

DELETE marks records in the active database file for deletion.

Syntax: DELETE (scope) FOR WHILE (condition)

Notes: Unless otherwise specified by the scope or the FOR WHILE clause, only the current record is marked for deletion.

With DISPLAY and LIST, records marked for deletion are indicated by an asterisk (*) in the first position of the record. Reinstate deleted records with RECALL.

With the full-screen commands, such as BROWSE or EDIT, records marked for deletion are indicated by *DEL* on the status line. In this mode, ^U both deletes and reinstates records.

DELETE does not reposition the record pointer. Therefore, if at the end of the file (EOF( ) = .T.), as after LIST or DISPLAY ALL, issuing DELETE has no effect.

Examples: The database Refnames from the Appendix is used for the following examples.

To mark only the first record in the database for deletion:

```
  USE Refnames
  DELETE
1 Record deleted
```

To mark record 10 for deletion:

```
  DELETE RECORD 10
1 Record deleted
```

See also: DELETED( ), PACK, RECALL, SET DELETED ON

DISPLAY

DISPLAY is used to view the contents of a database file

Syntax:

```
DISPLAY [OFF] [(scope)] [(expression list)]
        [FOR/WHILE (condition)]
        [TO PRINT]
```

Notes:

The current record is displayed unless otherwise specified by the scope or FOR/WHILE clause.

All fields are displayed unless otherwise specified.

The record number is displayed unless the OFF option is included.

If more than 20 records are displayed, DISPLAY pauses after every 20 lines. dBASE III prompts with "Press any key to continue . . ."

Each column has a heading. If the displayed item is the result of an expression involving a field (for example, cost*.25), the column heading is the same as the expression.

The heading appears as typed. For all capital letters, type DISPLAY FIELD1. For upper and lowercase headings, type DISPLAY Field1.

The contents of memo fields are not displayed unless explicitly named in the expression list. Instead, the field name appears above the word MEMO in the display. If memo fields are named in the expression list, their contents are displayed in 50 columns.

At the end of the file (EOF( ) = .T.), DISPLAY shows headings only, as there are no remaining records to display.

When DISPLAYing a record that is greater than 80 characters, the contents wrap-around to the next line and may break up a field onto two or more lines.

Example:

The following example uses the Rentals database from the Appendix.

Display the salesperson's initials, the monthly rental, and the percentage of commission times the monthly rental for the first five database records. The first two items are fields in the Rentals database. The last item is an expression involving two database fields.

```
  USE Rentals
. DISPLAY NEXT 5 Saleperson, Rent_month, ;
Perc_comm * Rent_month

Record#  Saleperson Rent_month Perc_comm * Rent_month
      1  GP             275.00               18.15000
      2  JS             800.00               60.80000
      3  DF             300.00               19.80000
      4  GP            3500.00              269.50000
      5  LG             850.00               56.95000
```

See also: SET HEADING

SORT

SORT creates a new database file in which the records of the active database file are reordered alphabetically, chronologically, or numerically by the specified key fields

Syntax:

```
SORT (scope) TO (newfile) ON (field) A C D
        (field2) A C D
        FOR WHILE (condition)
```

Notes:

The TO file has a .dbf extension unless otherwise specified

A file cannot be SORTed to itself or to any other open file

SORTs are in ascending order (A) unless otherwise specified. *Ascending order is the same as ASCII order* (see the Appendix)

The keywords ASCENDING and DESCENDING are alternatives for A and D

C causes the SORT not to differentiate between upper and lower case

You may combine C with either A or D When using two only include one slash, for example, DC

When SORTing on multiple fields, the most important key is placed first. Separate field names with commas

You may not SORT logical or memo fields

SORT does not work with substrings or complex expressions. To make these a key, use INDEX

Examples:

The database Rentals is to be SORTed on the rental agent's names. The field name is Saleperson

```
  USE Rentals
  SORT ON Saleperson TO Agent
100% Sorted            10 Records sorted
```

The same database Rentals is to be SORTed into rental comparisons. The fields involved are Saleperson and Rent_month. The SORTed file is to be put in order by Saleperson, then in descending numeric order by Rent_month.

```
  SORT ON Saleperson,Rent_month/D TO Agtrent
100% Sorted            10 Records sorted
```

See also: INDEX

USE

USE opens an existing database file and up to seven index files in the selected work area. If the database contains memo fields, the associated .dbt file is opened automatically.

Syntax:

```
USE <file name> INDEX <index file list>
    ALIAS <alias name>
```

Notes:

USE, without any parameters, closes the active database and index files in the currently selected work area.

Unless otherwise specified, dBASE III assumes a .dbf extension for the file name and .ndx extensions for the files in the index file list.

The index file list can consist of up to seven files if all of the files were created from the database file in USE.

If no alias name is included, the name given to the alias is the same as the database file name.

When a database file is USEd without any index files, the record pointer is positioned at the first record in the file.

When a database file is USEd with one or more index files, the record pointer is positioned at the first logical record *of the first index file listed. The record* pointer follows the order of that index file.

Examples:

To open the Refnames database without index files and set the alias to Refnames:

```
USE Refnames
```

To open this database with three index files and set the alias to F1:

```
USE Refnames INDEX Namelast,Namestat,Namezip ALIAS F1
```

See also: CLOSE, INDEX, SELECT, SET INDEX TO

Appendix B

Menu Handout

Database Structures

The address book maintenance database contains two databases. The address book database, called 'addbook' contains the name, address and telephone numbers of a number your friends. The other database, called 'xmaslist', contains information on who you sent Christmas cards to this past year and who sent one to you.

The name of the fields in each of the databases and their associated characteristics are described below. All fields are character data.

ADDBOOK

Name	Size	Comments
last_name	10	
first_name	10	
spouse_n	10	husband or wife first name
maiden_n	10	wife's maiden name
address	20	
city	14	
state	2	two letter abbreviation
zip	5	
area_code	3	
home_tel	8	format -> nnn-nnnn
work_tel	8	format -> nnn-nnnn

XMASLIST

Name	Size	Comments
last_name	10	same as last_name or maiden_n above
first_name	10	same as first_name or spouse_n above
sent	1	you sent this person a card (y/n)
received	1	you received a card from this person (y/n)

Menus

<u>Address Book Maintenance (Main-menu)</u>

This is the first menu in the address book maintenance program. The options that are available from this menu are:

1. Display the record format for each of the databases. Field names and their size are shown on the screen with an option to print.

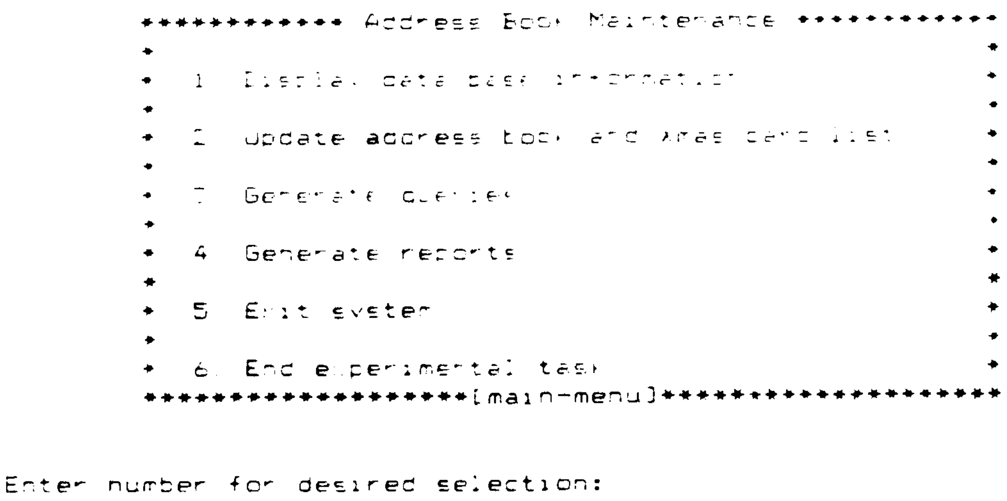
2. Through this option, you can add new entries to the databases, delete existing entries from the databases, or modify information already in the database.

3. This option will display information that meets specified criteria. The whole record(s) will be display.

4. Reports can be developed with this option. You can specify which fields to print in which order, sort order, data selection criteria and printing options.

5. This options leaves the maintenance system and returns to dBASE command level.

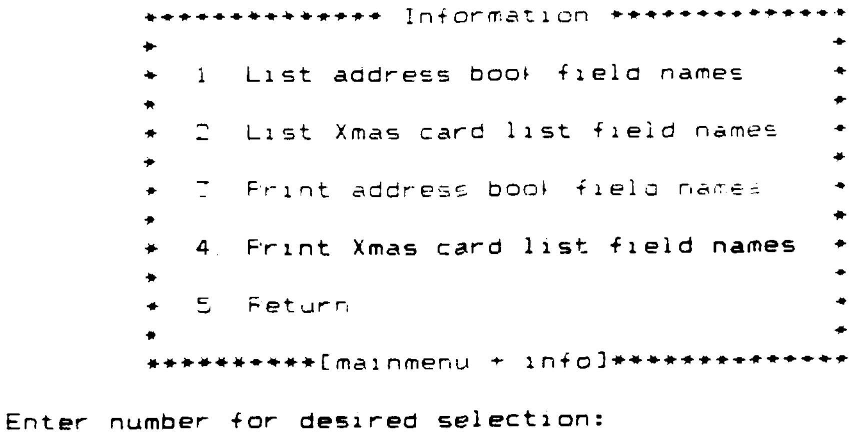
6. This option is used when you have finished a phase of the experiment. It just records the time that you finished a task.



```
************ Address Book Maintenance ************
*                                                *
*  1  Display data base information              *
*                                                *
*  2  Update address book and Xmas card list     *
*                                                *
*  3  Generate queries                           *
*                                                *
*  4  Generate reports                           *
*                                                *
*  5  Exit system                                *
*                                                *
*  6. End experimental task                      *
*******************[main-menu]*******************

Enter number for desired selection:
```

Information

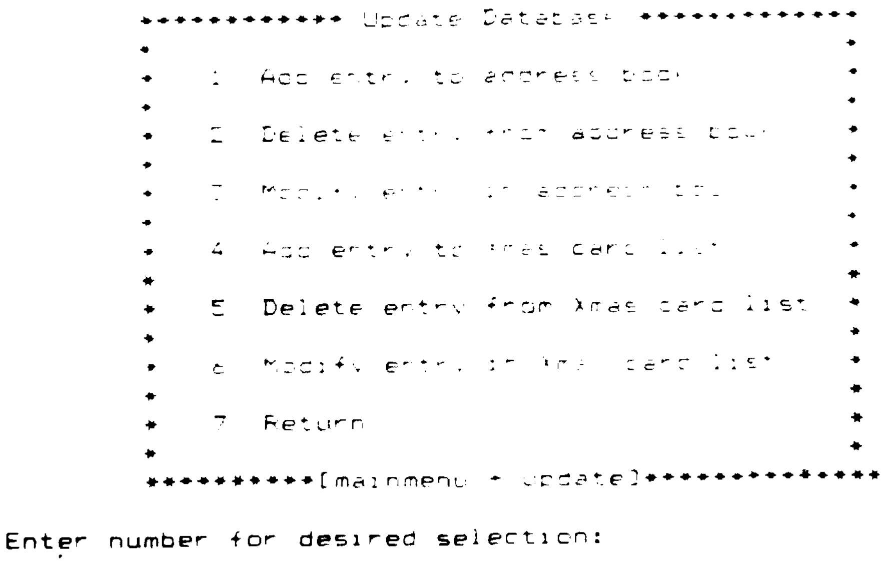
Option 1 from the Main-menu



```
************** Information *************
*                                      *
*  1  List address book field names    *
*                                      *
*  2  List Xmas card list field names  *
*                                      *
*  3  Print address book field names   *
*                                      *
*  4  Print Xmas card list field names *
*                                      *
*  5  Return                           *
*                                      *
**********[mainmenu + info]*************

Enter number for desired selection:
```

Update Database

Option 2 from the Main-menu. The first three option affect the address book database. Options 4 - 6 affect the Xmas card list. The last option returns you to the Main-menu. Only the address book menus will be shown since the Xmas card list menus are identical.

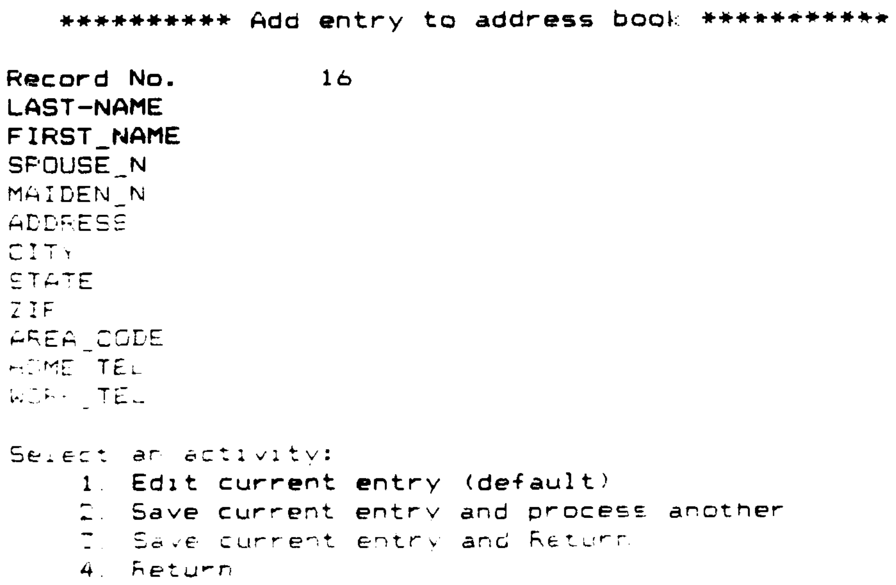


```
************* Update Database *************
*                                         *
*   1  Add entry to address book          *
*                                         *
*   2  Delete entry from address book     *
*                                         *
*   3  Modify entry in address book       *
*                                         *
*   4  Add entry to Xmas card list        *
*                                         *
*   5  Delete entry from Xmas card list   *
*                                         *
*   6  Modify entry in Xmas card list     *
*                                         *
*   7  Return                             *
*                                         *
**********[mainmenu + update]***************

Enter number for desired selection:
```

Add entry to address book

Option 1 from Update Database menu. All of the fields in the address book are displayed for you to enter information. The return key is used to move to the next field. The arrow control keys and the backspace key can be used to go back to a previous field. The options at the bottom of the screen appear after you have entered data into the fields. Null data (e.g. pressing return key without entering data) is allowed.

1. Allows you to go back and edit the information displayed.
2. Saves current information into the database and lets you create another database record.
3. Saves the current information into the database and returns you to the Update Database menu.
4. Returns you to the Update Database menu without saving the currently displayed information into the database.

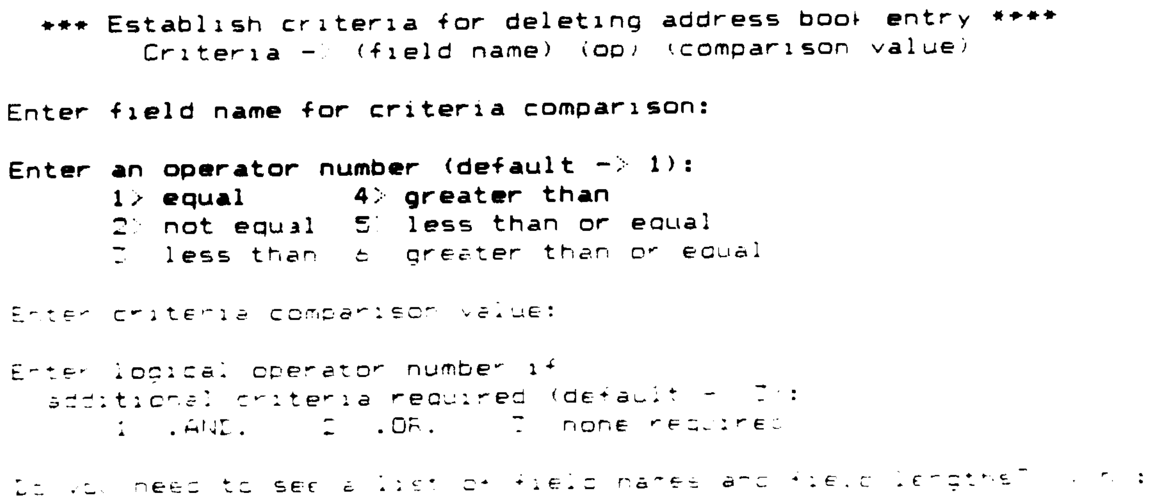


```
   *********** Add entry to address book ************

Record No.            16
LAST-NAME
FIRST_NAME
SPOUSE_N
MAIDEN_N
ADDRESS
CITY
STATE
ZIP
AREA_CODE
HOME_TEL
WORK_TEL

Select an activity:
     1. Edit current entry (default)
     2. Save current entry and process another
     3. Save current entry and Return
     4. Return
```

Establish criteria for deleting address book entry

Option 2 from the Update Database menu. This menu allows you to develop the criteria necessary for indicating which records are to be deleted. Once the criteria is established, the maintenance program will show you each record that meets the criteria so you can determine whether or not you want it deleted (see next menu)

Before entering criteria, you are given the option of looking at a list of the field names so you can spell them correctly. The criterion is specified by answering each of the four questions. The field name must be spelled correctly (it will be checked to be sure). Enter the number for the operator you want, not the operator itself. Enter the character string that you want to compare against. Only enter a logical operator if you need to specify more than one comparison requirement.



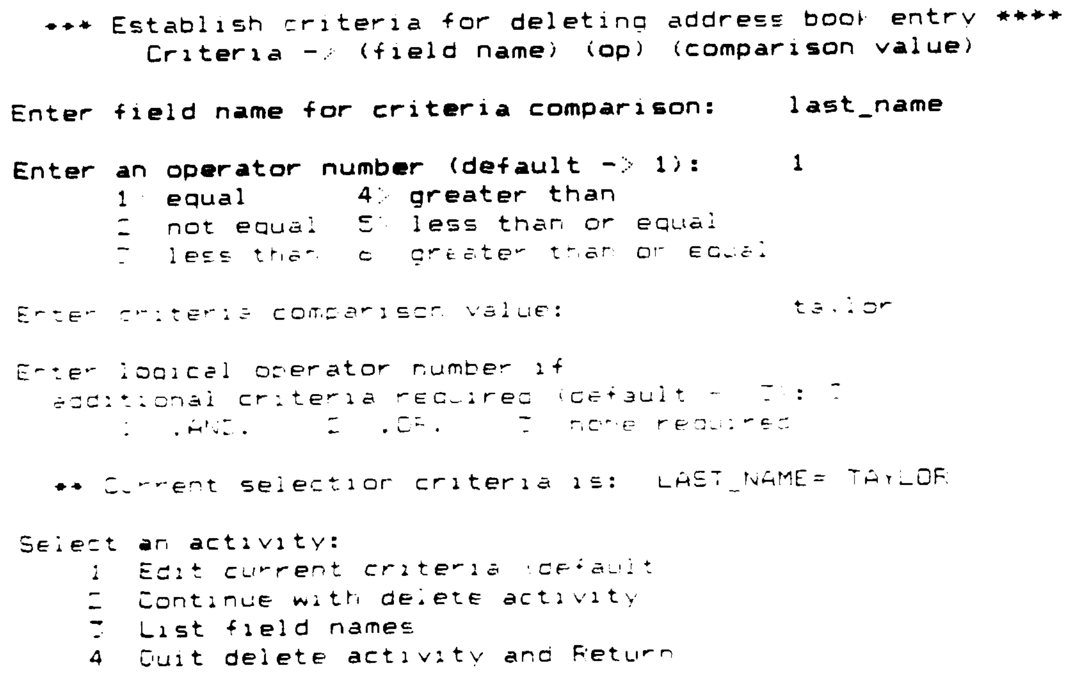
```
 *** Establish criteria for deleting address book entry ****
        Criteria -> (field name) (op) (comparison value)

Enter field name for criteria comparison:

Enter an operator number (default -> 1):
      1> equal       4> greater than
      2> not equal   5> less than or equal
      3> less than   6> greater than or equal

Enter criteria comparison value:

Enter logical operator number if
  additional criteria required (default -> 3):
      1> .AND.     2> .OR.     3> none required

Do you need to see a list of field names and field lengths? (Y/N):
```

For example, what if you wanted to delete all of the Taylors from your address book. After entering the criteria, the system will display the criteria back to you and present you with several options.

1. Edit the currently displayed criteria
2. Continue with the next phase of the delete activity.
    a. if you specified a logical operator it will let you enter next piece of the criteria.
    b. if no logical operator, it will go to the next phase of the delete activity.

3. List the names of the data fields (in case you forgot or got a name error) and allow you to edit the currently displayed criteria.

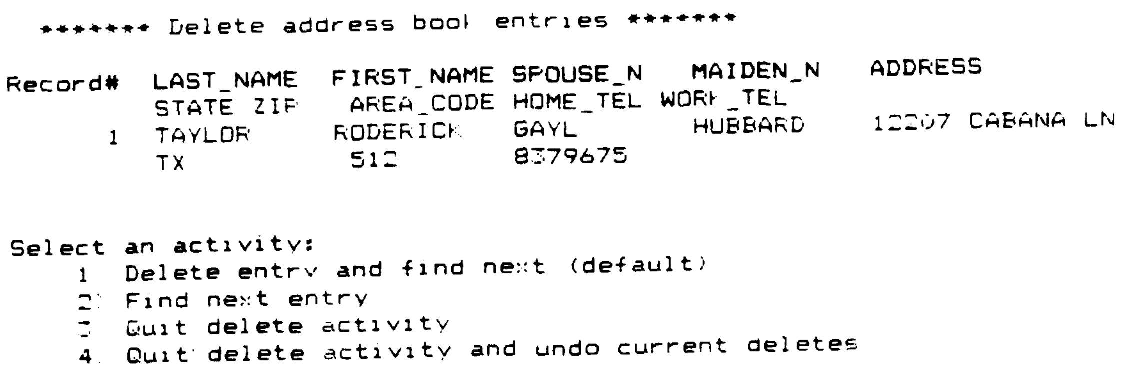
4. Quit what you are doing without doing anything and return to the Update Database menu.



```
   *** Establish criteria for deleting address book entry ****
        Criteria -> (field name) (op) (comparison value)

 Enter field name for criteria comparison:     last_name

 Enter an operator number (default -> 1):      1
        1) equal        4> greater than
        2) not equal    5) less than or equal
        3) less than    6) greater than or equal

 Enter criteria comparison value:              taylor

 Enter logical operator number if
   additional criteria required (default -> 3): 3
        1) .AND.     2) .OR.     3) none required

   ** Current selection criteria is:  LAST_NAME= TAYLOR

 Select an activity:
      1  Edit current criteria (default
      2  Continue with delete activity
      3  List field names
      4  Quit delete activity and Return
```

Delete address book entries

Option 2 from previous menu. The system will present each record that meet the delete criteria and let you perform any of the specified activities. When no further records meet the criteria, you have one last chance to undo your deletes before they become permanently deleted. You will return to the Update Database menu from here.



```
 ******* Delete address bool entries *******

Record#  LAST_NAME  FIRST_NAME SPOUSE_N   MAIDEN_N    ADDRESS
         STATE ZIP   AREA_CODE HOME_TEL WORK _TEL
     1   TAYLOR     RODERICK   GAYL        HUBBARD     12207 CABANA LN
         TX          512       8379675


Select an activity:
    1  Delete entrv and find next (default)
    2  Find next entry
    3  Quit delete activity
    4  Quit delete activity and undo current deletes
```

Establish criteria for modifying address book entry

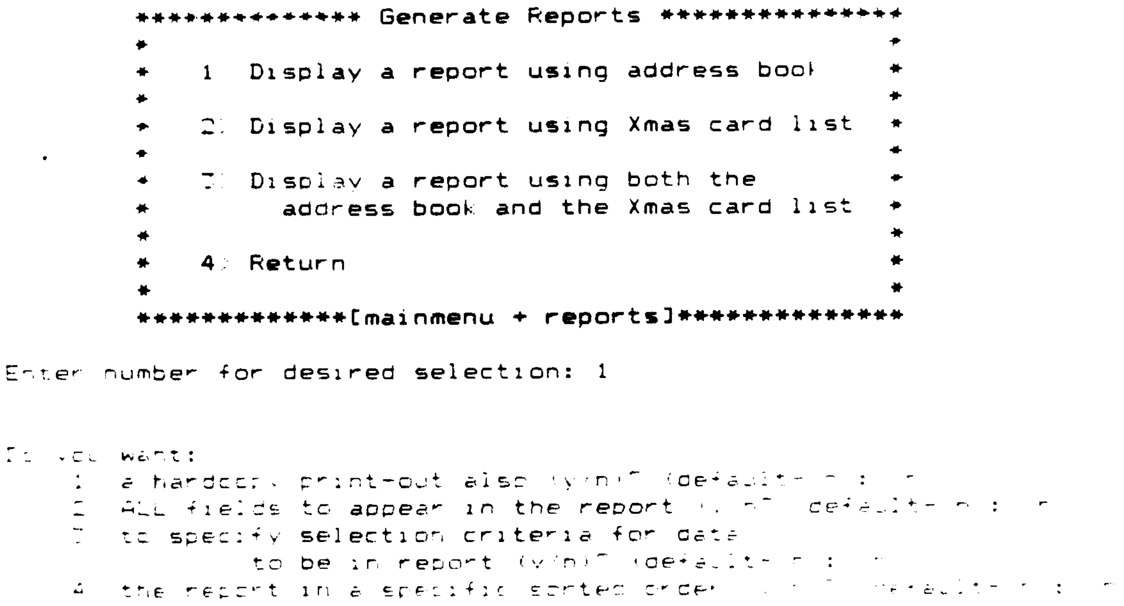
Option 3 from the Update Database menu. This menu is identical to the one used to establish the criteria for selecting records to be deleted. Records that meet the criteria are displayed in the same format as the Add record menu (with the current data in the record displayed instead of blanks) and are processed the same way. Options are self-explanatory.

Establish criteria for querying address book

Option 3 from Main-menu. This Menu is identical to the menu that establishes the delete record criteria. The "Continue with query activity" selection will display the record(s) that meet the criteria once you are through specifying the query criteria. After displaying the record(s), you will return to the Main-menu.

Generate Reports

Option 4 from the Main-menu. 1 and 2 generate reports from the specified database. Option 3 allows you to generate a report using data from both databases. Once you make a selection, you will be asked several additional questions that control the content and formatting of the report. After providing information in conjunction with these questions, a report will be displayed to the screen (optionally to the printer). You will return to this menu after the report is through displaying (printing).

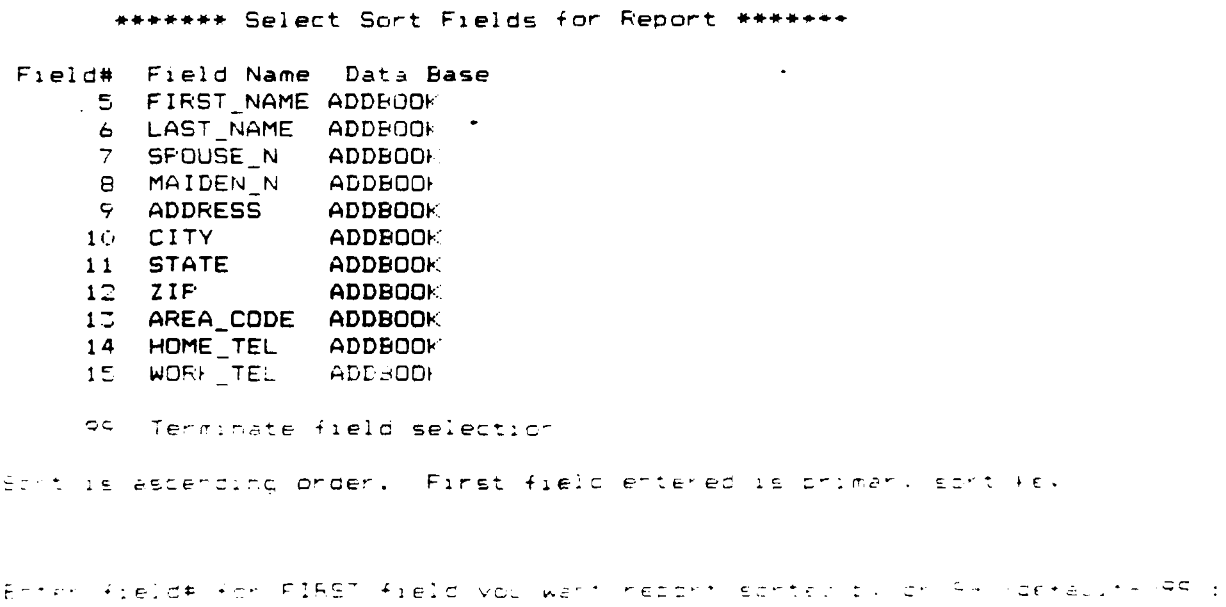


```
************** Generate Reports **************
*                                            *
*   1  Display a report using address book   *
*                                            *
*   2) Display a report using Xmas card list *
*                                            *
*   3) Display a report using both the       *
*        address book and the Xmas card list *
*                                            *
*   4) Return                                *
*                                            *
*************[mainmenu + reports]*************

Enter number for desired selection: 1


If you want:
    1  a hardcopy print-out also (y/n)? (default= n): n
    2  ALL fields to appear in the report (y/n)? (default= n): n
    3  to specify selection criteria for data
               to be in report (y/n)? (default= n): n
    4  the report in a specific sorted order (y/n)? (default= n): n
```

Select Sort Fields for Report

This menu will only be shown if you answer yes to the "sort order?" question. The fields that are available for sorting this report are displayed for you. You can sort on fields that are not selected as print fields in the report. The first field selected is the primary sort key with the others being secondary. Enter, one at a time, the number corresponding to the field you want to use as a sort key. Press return key after entering each field number.

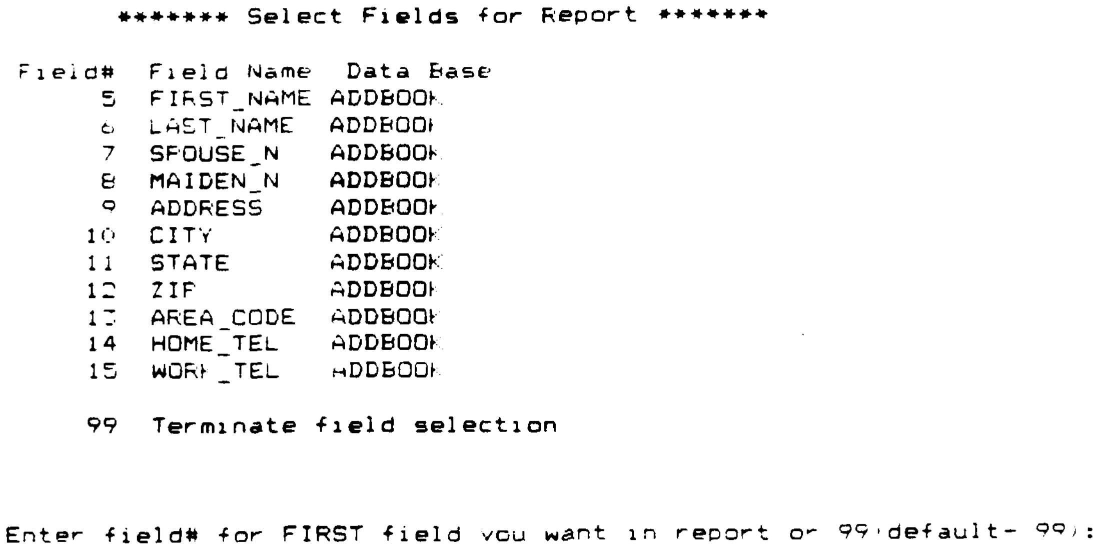
Once you have selected all the sort fields , enter '99' to terminate the selection process and go on to the next phase of the report generation process.



```
          ******* Select Sort Fields for Report *******

Field#  Field Name  Data Base
    5   FIRST_NAME  ADDBOOK
    6   LAST_NAME   ADDBOOK
    7   SPOUSE_N    ADDBOOK
    8   MAIDEN_N    ADDBOOK
    9   ADDRESS     ADDBOOK
   10   CITY        ADDBOOK
   11   STATE       ADDBOOK
   12   ZIP         ADDBOOK
   13   AREA_CODE   ADDBOOK
   14   HOME_TEL    ADDBOOK
   15   WORK_TEL    ADDBOOK

   99   Terminate field selection

Sort is ascending order.  First field entered is primary sort key.


Enter field# for FIRST field you want report sorted by, or 99 (default=99) :
```

Select Fields for Report

This menu is shown only if you do not want "ALL" of the available fields to be in the report. The fields that are available for displaying in this report are shown. Fields selected do not have to be the same as the sort fields. This menu is processed just like the sort field selection menu.



```
       ******* Select Fields for Report *******

Field#   Field Name  Data Base
     5   FIRST_NAME ADDBOOK
     6   LAST_NAME  ADDBOOK
     7   SPOUSE_N   ADDBOOK
     8   MAIDEN_N   ADDBOOK
     9   ADDRESS    ADDBOOK
    10   CITY       ADDBOOK
    11   STATE      ADDBOOK
    12   ZIP        ADDBOOK
    13   AREA_CODE  ADDBOOK
    14   HOME_TEL   ADDBOOK
    15   WORK_TEL   ADDBOOK

    99   Terminate field selection


Enter field# for FIRST field you want in report or 99(default= 99):
```

Establish report data selection criteria

This allows you to specify the criteria for selecting records to be in the report. The menu is identical to the Query criteria selection menu and processed the same way.

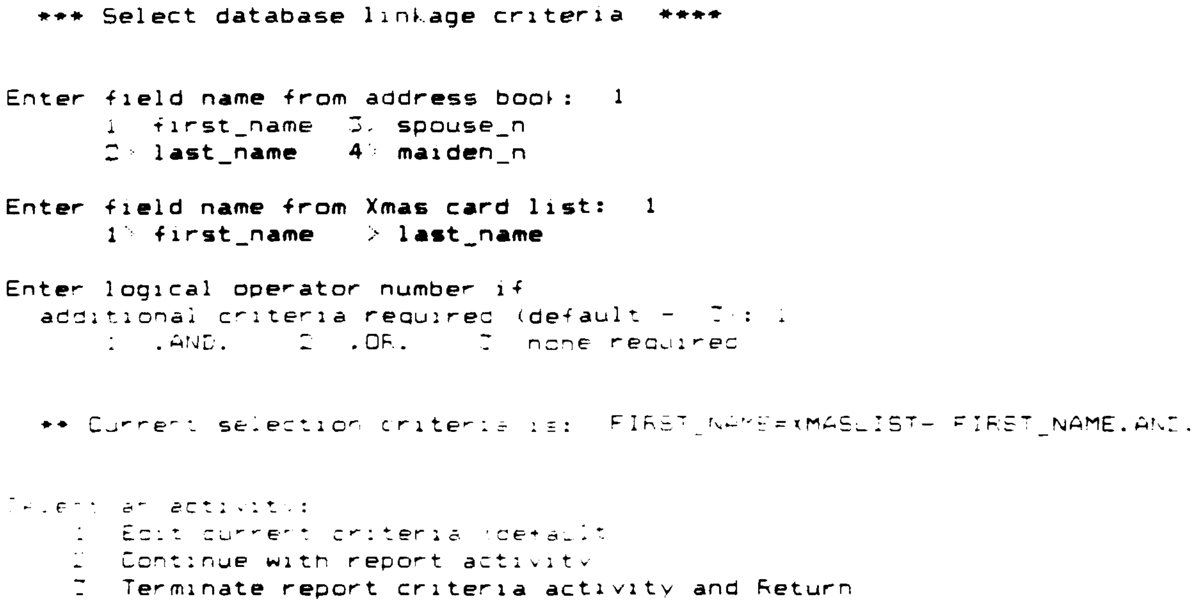
Select database linking criteria

This menu is displayed only if you select option 3 from the Generate Reports menu. This is used to match records between the two databases. To get a correct linkage, you need to specify a field that is a first name (first_name or spouse_n) and a field that is a last name (last_name or maiden_n) for each database. The options at the bottom of the screen are:

1. Allows you to edit current linkage criteria

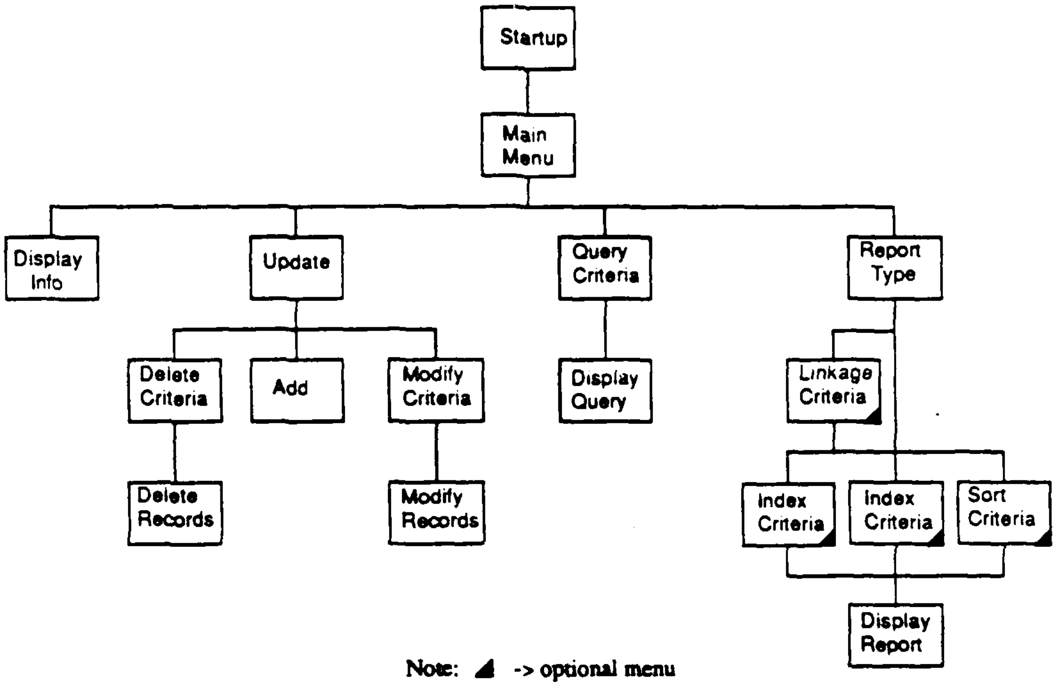
2. Continues with next linkage criteria if logical operator specified otherwise goes to next phase of report generation.

3. Returns you to the Generate Report menu without doing any further processing on this report.



```
  *** Select database linkage criteria  ****


Enter field name from address book:  1
      1) first_name  3) spouse_n
      2) last_name   4) maiden_n

Enter field name from Xmas card list:  1
      1) first_name   2) last_name

Enter logical operator number if
  additional criteria required (default - 3): 1
      1) .AND.    2) .OR.     3) none required


  ** Current selection criteria is:  FIRST_NAME=XMASLIST- FIRST_NAME.AND.


Select an activity:
    1) Edit current criteria (default)
    2) Continue with report activity
    3) Terminate report criteria activity and Return
```

Menu Dialog Mode Hierarchy



Note: ◢ -> optional menu

Appendix C

Experimental Tasks

Task Scoring

1. Task 1 consisted of 8 numbered items that contained 13 actions that were scored for a total of 52 points maximum.

2. Task 2 consisted of 6 numbered items that contained 10 actions that were scored for a total of 40 points maximum.

3. The scoring for performance effeciveness was based on an iterval scale from 1 to 4.
   4 - Completely correct
   3 - almost correct (minor discrepancy)
   2 - partially correct (major discrepancy)
   1 - totally incorrect or not done

Total maximum points available for a raw performance score was 92. A subject's raw score was then devided by the number of actions (23) to get the average score per action that represented a performance effectiveness score.

4. The raw score for performance efficiency was the total time to perform all of the actions on the two tasks. The maximum time was 45 minutes per task for a total of 90 minutes. The number was then divided by the number of actions completed to get the average time to complete an action that represented the efficiency score.

Task 1

It has been over three months since Christmas and you have decided to finally update your address book and Christmas card list with all the changes that have occurred. Being an orderly person, you have made some notes on the things that need to be done and plan to do then in the order you have them listed. You have allotted yourself 45 minutes to try and get as much done as possible.

1. You sent both Pete Zack and Wanda Wonder cards this year but forgot to put then in your Xmas card list and now would be a good time to add them. They both sent you a card in return.

2. You got a card from John and Mary Able this year and you know that you didn't send them one. You are not even sure if they are still on your Xmas card list. Check to see if they are on the list. If not, add them to the list. If they are still on the list, just update the information.

3. Make the following changes to the address book:
   a. The Baker's new home telephone number is 555-2345.
   b. Linda Floatsome now lives at:
      1437 Beale St.
      Austin, Tx 78727
   c. Chuck Meat has re-married again. This time it is to someone named "Bootsie".

4. You want to cut back on you Xmas card list so display all of the people who you didn't receive a card from. You'll want to make a hardcopy print-out of the result for future reference because you are also going to delete only those people who you didn't send a card to <u>and</u> you didn't receive a card from them either.

5. The area code for your hometown of Brady TX was changed from 512 to 915. You need to change this for all of your friends in Brady. List all of the people who live in Brady when you are finished with the changes to be sure they are all correct (list first names, last names, city, state and area code).

6. Make a list of all the Hadleys and Taylors. Only print-out their last name, first name, city and state. List them so they are ordered by state then city.

7. Two of your good friends, John Ricket and Brenda Smith got married over Christmas. They now live at Brenda's house. Combine these two together into one address book entry. Don't forget to get rid of the entry(s) no longer needed. List Brenda as the spouse. When you are finished with the update, list the new entry to be sure everything is ok.

8. You plan to drop a quick note to all those people from whom you received a card but you didn't send them one. First list them with sent and received shown. Then make a report that lists the names and addresses of these people. Do not worry about sorting it. You need to link (join) the two databases for this report.

Task 2

You just remembered another set of notes that you had stuffed into a desk drawer a few weeks back. Since you have the system up and running you decide to go ahead and do the work. This time, after 45 minutes you'll really quit and go do something else.

(PLEASE DO ITEM '6' ON MAINMENU OR 'DO ENDITEM' WITH COMMAND LANGUAGE AFTER YOU COMPLETE EACH NUMBERED ITEM BELOW)

1. You noted that you misspelled a couple of your friends names. Their last name is "Wiederman" and not "Weiderman". You need to correct this in both databases. Display the corrections when finished.

2. You have been meaning to call an old friend of yours, Karen Issette. but you can't remember her husband's name (don't want to be embarrassed if he remembers yours). List her husbands name and their phone number.

3. Your Mama wants you to give her the names and addresses of the Smith girls now that they are all married. She wants to send them all a little something. List both their names and their husband's name plus their mailing address.

4. The post office left you a little card about a month ago saying that all 78727 zip codes are changed to 78759 because a new post office has opened. See if any of your friends have a 78727 zip and change them to the new one.

5. That was the last thing you really needed to do but you decide to just "poke around" in the data bases a little longer before making a backup of you changes.

   a. List all of your married friends, husband and wife, sorted by city.
   (first name, spouse name, last name,city, and state)

   b. Make a list of everyone that was sent an Xmas card and doesn't live in TX. (first name, last name, city, state, sent)

   c. List everyone who doesn't have a home or work phone listed in the address book. You plan to try and get them one of these days. (last name, first name, home and work telephone numbers)

6. Now that you are through fooling around, you need to make a list (sorted by last name) of everybody in the address book (all fields) as your backup. Do this for the Xmas card list as well.

Appendix D

Computer Technology Experience Survey

COMPUTER TECHNOLOGY EXPERIENCE SURVEY

Thank you for volunteering to participating in this experiment. This survey will be used to classify individuals into several groups from which subjects will be randomly selected for the experiment. You will be contacted within the next week if you are selected. The last four digits of your social security number (or student id number) are only needed for tracking subjects through the experiment as part of a group. Your name and phone number are needed to contact you about the experiment.

Name_________________________ Telephone_____________

SSAN (last 4 digits)______

1. List approximate number of undergraduate classes where using a computer was an integral part of the course requirement (e.g. programming class, data base class, statistics class). _____

2. How many of those undergraduate classes listed above were data processing or computer science courses? _____

3. List approximate number of graduate classes where using a computer was an integral part of the course requirement (e.g. programming class, data base class, statistics class). _____

4. How many of those graduate classes listed above were data processing or computer science courses? _____

5. How many years/months of work experience do you have with computer technology (such as programming, data base, teaching, and design)?
_____years _____months

6. Which of the following programming languages have you used for at leaste three months or had as a semester course?

__Cobol __Fortran __Basic __PL/I __Lisp __Assembler __Other

7. Which of the following type of computer equipment have you worked with?

__Personal computer __IBM mainframe __VAX mainframe
__CDC mainframe __Mini computer __Other mainframes

8. Which of the following types of computer applications have you had experience with?

__Database (e.g. dBASE, Ingres,System 2000)
__Spread sheet (e.g. Lotus 1-2-3, Multiplan)
__Word processing (e.g. Wordstar, XEDIT, Word,Script)
__Statistical packages (e.g. SAS, IMS)
__Electronic mail

9. How much dBASE experience have you had?

__None
__Introduction (less than 3 hrs instruction)
__Introduction plus completed a small exercise (e.g. DPA 385 exercise)
__Training (over 3 hrs instruction) plus several small exercises
__Use occasionally but have developed several small and large projects
__Use regularly to develop both large and small systems

10. How long have you been using dBASE (days, weeks or years)? _____

11. In your opinion, how would you classify youself on your overall dBASE experience and knowledge?

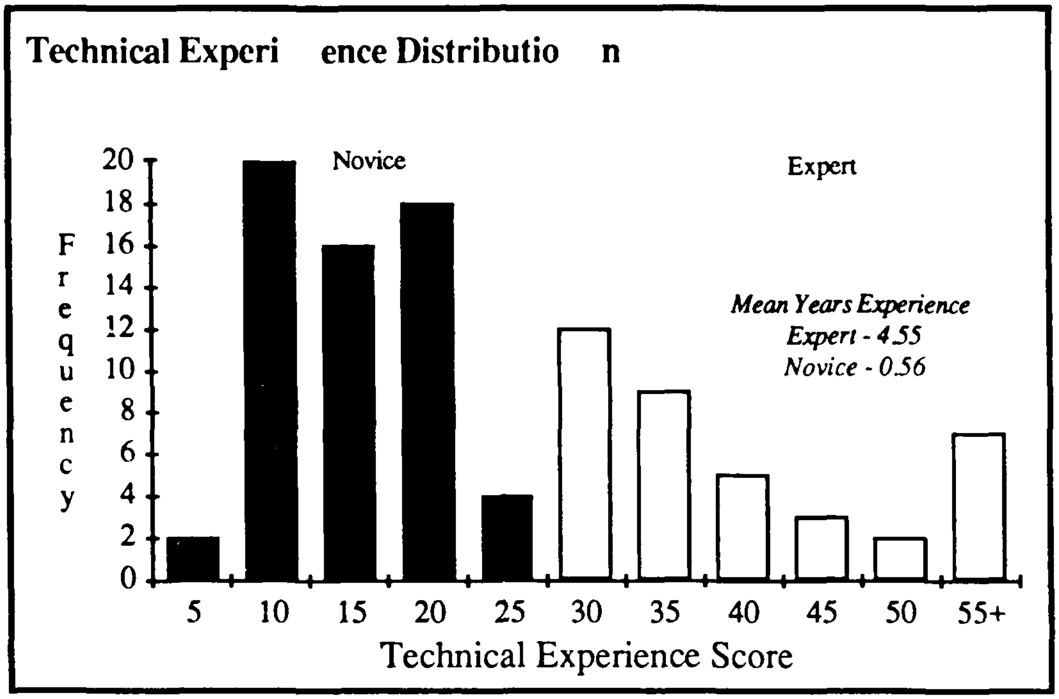
__No experience __Novice __Intermediate
__Advanced __Very experienced __Expert

12. In your opinion, how would you classify youself on your overall computer experience and knowledge?

__No experience __Novice __Intermediate
__Advanced __Very experienced __Expert

Times that you would **not** be available for participating in the experiment

	Morning (9 - 11)	Early Afternoon (1 - 3)	Late afternoon (4:30 - 6:30)	Evening (7 - 9)
Mon				
Tue				
Wed				
Thur				
Fri				
Sat				



Technical Experience Distribution

Mean Years Experience
Expert - 4.55
Novice - 0.56

Appendix E

Satisfaction Survey

SATISFACTION SURVEY

1. Consider your feeling about the computer interface mode (e.g. menu, command language, etc) used in this experiment. Indicate where you feelings best lie between these pairs of contrasting words.

Extremely Quite Slightly Neutral Slightly Quite Extremely

Easy-to-use :______:______:______:______:______:______:______: Hard-to-use

Frustrating :______:______:______:______:______:______:______: Comfortable

Simple :______:______:______:______:______:______:______: Complicated

Hard-to-learn :______:______:______:______:______:______:______: Easy-to-learn

Confusing :______:______:______:______:______:______:______: Obvious

Satisfying :______:______:______:______:______:______:______: Dissatisfying

2. Consider these statements and indicate where you feelings lie between the pairs of words. The range of values are the same as used in question 1 above.

To me, the type of computer interface mode that a computer system uses is:

Important :______:______:______:______:______:______:______: Unimportant

Overall, my satisfaction with the computer interface mode used in this experiment is:

Satisfactory :______:______:______:______:______:______:______: Unsatisfactory

Overall, my satisfaction with the task required in this experiment (updating address book and Christmas card list) is:

Unsatisfactory :______:______:______:______:______:______:______: Satisfactory

Overall, my satisfaction with this experiment is:

Unsatisfactory :______:______:______:______:______:______:______: Satisfactory

3. Consider the following statements. How much do you agree or disagree with each? We're interested in your impressions about computer interface modes in general and your impressions about the computer interface mode used in this experiment.

a. I prefer a computer system to use Menus rather than a Command Language as the computer interface mode.

Strongly Agree	Agree	Slightly Agree	Neutral	Slightly Disagree	Agree	Strongly Disagree

b. I prefer a computer system to have more than one computer interface mode (both menu and command language for example).

Strongly Agree	Agree	Slightly Agree	Neutral	Slightly Disagree	Agree	Strongly Disagree

4. Please feel free to write any comments you want about the experiment as a whole, the task you were asked to perform, the computer dialog mode used, the handouts used in the experiment, the training or anything else. Thanks again for participating.

Appendix F

Sample Subject Task Logs

Sample Log Scoring

Two partial logs were selected to provide a sample of the type of output generated by the experiment. The logs are the first four items in task two. The handwriting in the logs is the scorer's marks for completing each item. Both logs have a dump of the data base at the end for the scorer to use in determining the correctness of adds, deletes, and modification requirements. No judgements were made on how the subject completed each item. They were scored strictly on whether or not they accomplished what was required; thus, there was quite a variety of methods used to complete the items in the tasks.

The command language subject, ID = 8350, was an expert in technology but had no dBASE experience. This subject received full credit for each item part scored. In the #3 item, only two individuals were listed for the correctly stated DISPLAY statement instead of 3 because the subject did not add the third person as part of an item in task 1. The log reflects everything typed by the subject even those characters that were erased with the backspace character.

The menu subject, ID = 8224, was also an expert in technology without any previous dRASE experience. This subject received a +4,+0 on item #1 because the subject correctly made the changes (+4) but did not list the changes as required in the item; thus, a +0. Full credit was obtained on the other items. The menu log consisted of at leaste one line of text from each menu presented to the subject, used to track the subject's progress, and the minimum amount of data, entered by the subject on the menu, required to determine his actions.

Sample Command Language Log

```
. UDSE ADDBOOK                        #1

. REPLACE LAST_NAME WITH 'WIEDERMAN' FOR LAST-NAME_NAME = 'WEIDERMAN'

        2 records replaced   ✓
. USE XMANSLIST

. REPLACE LAST-NAME _NAME WITH 'WIEDERMAN' FOR LAST_NAME = 'WEIDERMAN'

        2 records replaced   ✓          +4
. DISPLAY FOR LAST_NAME='WEIIEDERMAN'

Record#  LAST_NAME  FIRST_NAME SENT RECEIVED
     16  WIEDERMAN  RICHARD     Y     N      ✓
     17  WIEDERMAN  GERTRUDE    N     Y
. USE ADDBOOK

. DISPLAY FOR LAST_NAME='WIEDERMAN'
                                                          +4
Record#  LAST_NAME  FIRST_NAME SPOUSE_N   MAIDEN_N   ADDRESS
CITY     STATE ZIP   AREA_CODE HOME_TEL WORK_TEL
     22  WIEDERMAN  RICHARD                          3428 ACORN  ✓
HOUSTON  TX     79884 809       345-9978
     23  WIEDERMAN  GERTRUDE                         1834 WALNUT
HOUSTON  TX     78922 432       332-8999 332-8476
. DO ENDITEM

04/15/86
02:00:01

Enter item number you just finished: 1
--------------------------------------------------
                                             #2
If last item in task, tell monitor
Press any key to continue...
. DISPLAY FOR LAST_NAME='ISSETTE'.AND.FIRST_NAME='KAREN' SPOUSE_N,
HOME_TEL

Record#  SPOUSE_N   HOME_TEL

. DISPLAY FOR LAST_NAME='ISSETTE'.AND.(FIRST-NAME='KAREN .OR.
SPOUSE_N='KAREN') SPOUSE_N, HOME_TEL

Variable not found
                                                ?
DISPLAY FOR LAST_NAME='ISSETTE'.AND.(FIRST-NAME='KAREN'.OR.
SPOUSE_N='KAREN') SPOUSE_N, HOME_TEL
Do you want some help? (Y/N) No
. DISPLAY FOR LAST_NAME='ISSETTE'.AND.(FIRST_NAME='KAREN'.OR.
SPOUSE_N='KAREN') SPOUSE_N, HOME_TEL

Record# SPOUSE_N   HOME_TEL
      2 KAREN      445-9483
. DISPLAY

Record# LAST_NAME  FIRST_NAME SPOUSE_N   MAIDEN_N   ADDRESS
CITY      STATE ZIP   AREA_CODE HOME_TEL WORK_TEL
. LOCATE LAST_NAME='ISSETTE'

Syntax error
       ?
LOCATE LAST_NAME='ISSETTE'
Do you want some help? (Y/N) No
. LOCATE FOR LAST_NAME='ISSETTE'

Record =      2
. DISPLAY

Record# LAST_NAME  FIRST_NAME SPOUSE_N   MAIDEN_N   ADDRESS
CITY      STATE ZIP   AREA_CODE HOME_TEL WORK_TEL
      2 ISSETTE    JOSE       KAREN      SWOOP      2342 WEST CIRCLE
AUSTIN    TX    78727 512        445-9483 471-3322
. DISPLAY FIRST_NAME,HOME_TE.L

Record# FIRST_NAME HOME_TEL
      2 JOSE       445-9483      +4
. DOE ENDITEM

04/15/86
02:04:48

Enter item number you just finished: 2

If last item in task, tell monitor      #3
Press any key to continue...
. DISPLAY FOR MAIDEN_N='SMITH' SPOUSE_N, FIRST_NAEMEAME, ADDRESS, CITY,
STATE,ZIP

Record# SPOUSE_N   FIRST_NAME ADDRESS      CITY       STATE ZIP
      5 GRETCH     BILL       934 JOJO     FORT WORTH TX    75664
      8 BETTY      RICHARD    4434 4TH ST BRADY       TX    78725
```

```
. DISPLAY FOR MAIDEN_N='SMITH' SPOUSE_N, FIRST-NAME,LAST)_N,
LASTLAST_NAME,FIRST_NAME, ADDRESS,CITY,STAT,ETE,TZIP

Record#  SPOUSE_N   LAST_NAME   FIRST_NAME ADDRESS     CITY       STATE
ZIP
      5  GRETCH     HOLMES      BILL       934 JOJO    FORT WORTH TX      +4
75664
      8  BETTY      KELLY       RICHARD    4434 4TH ST BRADY      TX
78725
. DO ENDITEM

04/15/86
02:09:14

Enter item number you just finished: 3
                                            #4
If last item in task, tell monitor
Press any key to continue...
. REPLACE ZIP WITH 'U8759' F078759' FOR ZIP='78727

Unterminated string
                                            ?
REPLACE ZIP WITH '78759' FOR ZIP='78727
Do you want some help? (Y/N) No
REPLACE ZIP WITH '78759' FOR ZIP='78727' ✓

        5 records replaced
. DISPLAY FOR ALL WITHZIP='78759'

Record#  ZIP='78759'
      1  .T.
      2  .T.
      3  .F.
      4  .F
      5  .F.
      6  .F.
      7  .F.
      8  .F.
      9  .F.
     10  .F.
     11  .T.
     12  .T.
     13  .F.
     14  .F.
     15  .F.
     16  .F.
     17  .F.
     18  .T.
     19  .F.
     20  .F.
Press any key to continue
Record#  ZIP='78759'
     21  .F.
     22  .F.
     23  .F.
     24  .F.
.  DISPLAY FOR ZIP='78759'

Record#  LAST_NAME  FIRST_NAME SPOUSE_N   MAIDEN_N   ADDRESS
CITY     STATE ZIP   AREA_CODE HOME_TEL WORK_TEL
      1  TAYLOR     RODERICK   GAYL       HUBBARD    12207 CABANA LN
AUSTIN   TX    78759 512       837-9675 471-3322
      2  ISSETTE    JOSE       KAREN      SWOOP      2342 WEST CIRCLE
AUSTIN   TX    78759 512       445-9483 471-3322.                      ✓
     11  TAYLOR     HARRY      CAROLYN    LEWIS      12602 TERRA NOVA   +4
AUSTIN   TX    78759 512       555-9987
     12  HADLEY     TRENT      MARTHA                56707 CAVERN SPGS
AUSTIN   TX      78759
     18  FLOATSOME  LINDA                            1437 BEALE ST.
AUSTIN   TX      78759

.  DO ENDITEM

04/15/86
02:11:46

Enter item number you just finished: 4

if last item in task, tell monitor
Press any key to continue...
```

Sample Menu Log

```
04/14/86
01:31:38
        ****************[main-menu]****************
Enter number for desired selection: 2

        ****************[mainmenu + update]****************

Enter number for desired selection: 3

  *** Establish criteria for modifying address book entry ****

Do you need to see a list of field names and field lengths? (y/n): N

  *** Establish criteria for modifying address book entry ****

  ** Current selection criteria is:  LAST_NAME='WEIDERMAN'
Select an activity: 2

  ********** Modify entry address book *********
Record No.          21
Select an activity: 3

        ****************[mainmenu + update]****************

Enter number for desired selection: 3

  *** Establish criteria for modifying address book entry ****

Do you need to see a list of field names and field lengths? (y/n): N

  *** Establish criteria for modifying address book entry ****

  ** Current selection criteria is:  LAST_NAME='WEIDERMAN'
Select an activity: 2

  ********** Modify entry address book *********
Record No.          22
Select an activity: 2

...End of database reached.  No records left to examine.
Press any key to continue...K
        ****************[mainmenu + update]****************

Enter number for desired selection:
```

```
Enter number for desired selection: 6


  *** Establish criteria for modifying Xmas card list entry ****

Do you need to see a list of field names and field lengths? (y/n): N

  *** Establish criteria for modifying Xmas card list entry ****

  ** Current selection criteria is:  LAST_NAME='WEIDERMAN'
Select an activity: 2

  ********** Modify entry Xmas card list *********
Record No.         19
Select an activity: 2
                                            ✓

Record No.         20
Select an activity: 3                                   +4
                                                        +ø

          ****************[mainmenu + update]****************

Enter number for desired selection: 7

          ****************[main-menu]****************
Enter number for desired selection: 6

04/14/86
01:36:59
Enter item number you just finished: 1
                                                  #2
If last item in task, tell monitor
Press any key to continue...K
          ****************[main-menu]****************
Enter number for desired selection: 4

          ****************[mainmenu + reports]****************

Enter number for desired selection: 1

  *** Establish addbook report data selection criteria ****

Do you need to see a list of field names and field lengths? (y/n):N

  *** Establish addbook report data selection criteria ****
```

```
 ** Current selection criteria is:  LAST_NAME='ISSETTE'
Select an activity: 2

          ******* Select Fields for Report *******
          5  FIRST_NAME ADDBOOK
          6  LAST_NAME  ADDBOOK
          7  SPOUSE_N   ADDBOOK
          8  MAIDEN_N   ADDBOOK
          9  ADDRESS    ADDBOOK
         10  CITY       ADDBOOK
         11  STATE      ADDBOOK
         12  ZIP        ADDBOOK
         13  AREA_CODE  ADDBOOK
         14  HOME_TEL   ADDBOOK
         15  WORK_TEL   ADDBOOK
Enter field# for FIRST field you want in report or 99(default->99): 5

Current field list: FIRST_NAME
Enter field# for NEXT field you want in report or 99(default->99): 13

Current field list: FIRST_NAME,AREA_CODE
Enter field# for NEXT field you want in report or 99(default->99): 14

Current field list: FIRST_NAME,AREA_CODE,HOME_TEL
Enter field# for NEXT field you want in report or 99(default->99): 99

  FIRST_NAME AREA_CODE HOME_TEL
  JOSE       512       445-9483 ✓ +4
Press any key to continue...K
        ****************[mainmenu + reports]****************

Enter number for desired selection: 4

        ****************[main-menu]****************
Enter number for desired selection: 6

04/14/86
01:38:47
Enter item number you just finished: 2
                                             #3
If last item in task, tell monitor
Press any key to continue...K
        ****************[main-menu]****************
Enter number for desired selection: 4
```

```
        *****************[mainmenu + reports]*****************

Enter number for desired selection: 1

  *** Establish addbook report data selection criteria ****

Do you need to see a list of field names and field lengths? (y/n):N

  *** Establish addbook report data selection criteria ****

  ** Current selection criteria is:  MAIDEN_N='SMITH'
Select an activity: 2

          ******* Select Fields for Report *******
          5  FIRST_NAME ADDBOOK
          6  LAST_NAME  ADDBOOK
          7  SPOUSE_N   ADDBOOK
          8  MAIDEN_N   ADDBOOK
          9  ADDRESS    ADDBOOK
         10  CITY       ADDBOOK
         11  STATE      ADDBOOK.
         12  ZIP        ADDBOOK
         13  AREA_CODE  ADDBOOK
         14  HOME_TEL   ADDBOOK
         15  WORK_TEL   ADDBOOK
Enter field# for FIRST field you want in report or 99(default->99): 7

Current field list: SPOUSE_N
Enter field# for NEXT field you want in report or 99(default->99): 5

Current field list: SPOUSE_N,FIRST_NAME
Enter field# for NEXT field you want in report or 99(default->99): 6

Current field list: SPOUSE_N,FIRST_NAME,LAST_NAME
Enter field# for NEXT field you want in report or 99(default->99): 9

Current field list: SPOUSE_N,FIRST_NAME,LAST_NAME,ADDRESS
Enter field# for NEXT field you want in report or 99(default->99): 10

Current field list: SPOUSE_N,FIRST_NAME,LAST_NAME,ADDRESS,CITY
Enter field# for NEXT field you want in report or 99(default->99): 11

Current field list: SPOUSE_N,FIRST_NAME,LAST_NAME,ADDRESS,CITY,STATE
Enter field# for NEXT field you want in report or 99(default->99): 12
```

```
Current field list: SPOUSE_N,FIRST_NAME,LAST_NAME,ADDRESS,CITY,STATE,ZIP
Enter field# for NEXT field you want in report or 99(default->99): 99

SPOUSE_N   FIRST_NAME LAST_NAME  ADDRESS       CITY        STATE  ZIP
GRETCH     BILL       HOLMES     934 JOJO      FORT WORTH  TX     75664  ✓
BETTY      RICHARD    KELLY      4434 4TH ST   BRADY       TX     78725
BRENDA     JOHN       RICKET     826 WEST 10TH FORT WORTH  TX     76744  ✓4
Press any key to continue...K
        ****************[mainmenu + reports]****************

Enter number for desired selection: 4

        ****************[main-menu]****************
Enter number for desired selection: 6

04/14/86
01:40:46
Enter item number you just finished: 3
                                                #4
If last item in task, tell monitor
Press any key to continue...K
        ****************[main-menu]****************
Enter number for desired selection: 24

        ****************[mainmenu + reports]****************

Enter number for desired selection: 1

  *** Establish addbook report data selection criteria ****

Do you need to see a list of field names and field lengths? (y/n):N

  *** Establish addbook report data selection criteria ****

  ** Current selection criteria is:  ZIP='78727'
Select an activity: 2

          ******* Select Fields for Report *******
        5  FIRST_NAME ADDBOOK
        6  LAST_NAME  ADDBOOK
        7  SPOUSE_N   ADDBOOK
        8  MAIDEN_N   ADDBOOK
        9  ADDRESS    ADDBOOK
       10  CITY       ADDBOOK
       11  STATE      ADDBOOK
```

```
        12  ZIP         ADDBOOK
        13  AREA_CODE   ADDBOOK
        14  HOME_TEL    ADDBOOK
        15  WORK_TEL    ADDBOOK
Enter field# for FIRST field you want in report or 99(default->99): 6

Current field list: LAST_NAME
Enter field# for NEXT field you want in report or 99(default->99): 5

Current field list: LAST_NAME,FIRST_NAME
Enter field# for NEXT field you want in report or 99(default->99): 12

Current field list: LAST_NAME,FIRST_NAME,ZIP
Enter field# for NEXT field you want in report or 99(default->99): 99

 LAST_NAME  FIRST_NAME ZIP
 TAYLOR     RODERICK   78727
 ISSETTE    JOSE       78727
 TAYLOR     HARRY      78727
 HADLEY     TRENT      78727
 FLOATSOME  LINDA      78727
press any key to continue...K
        ****************[mainmenu + reports]****************

Enter number for desired selection: 4

        ****************[mainmenu + update]****************

Enter number for desired selection: 3

  *** Establish criteria for modifying address book entry ****

Do you need to see a list of field names and field lengths? (y/n). N

  *** Establish criteria for modifying address book entry ****

  ** Current selection criteria is:  LAST_NAME='TAYLOR'
Select an activity: 2

  ********** Modify entry address book *********
Record No.          1
Select an activity: 3

        ****************[mainmenu + update]****************
```

```
Enter number for desired selection: 3

  *** Establish criteria for modifying address book entry ****

Do you need to see a list of field names and field lengths? (y/n): N

  *** Establish criteria for modifying address book entry ****

  ** Current selection criteria is:  LAST_NAME='ISSETTE'
Select an activity: 2

  ********** Modify entry address book *********
Record No.          2
Select an activity: 3

        ****************[mainmenu + update]****************

Enter number for desired selection: 3

  *** Establish criteria for modifying address book entry ****

Do you need to see a list of field names and field lengths? (y/n): N

  *** Establish criteria for modifying address book entry ****

  ** Current selection criteria is:  LAST_NAME='TAYLOR'
Select an activity: 2

  ********** Modify entry address book *********
Record No.          1
Select an activity: 2

Record No.          10
Select an activity: 3

        ****************[mainmenu + update]****************

Enter number for desired selection: 3

  *** Establish criteria for modifying address book entry ****

Do you need to see a list of field names and field lengths? (y/n): N

  *** Establish criteria for modifying address book entry ****
  ** Current selection criteria is:  LAST_NAME='HADLEY'
Select an activity: 2

  ********** Modify entry address book *********
Record No.         3
Select an activity: 4
                         ✓
Record No.         11
Select an activity: 3

        ****************[mainmenu + update]****************

Enter number for desired selection: FLOATSOME3

  *** Establish criteria for modifying address book entry ****

Do you need to see a list of field names and field lengths? (y/n): N

  *** Establish criteria for modifying address book entry ****

  ** Current selection criteria is:  LAST_NAME='FLOATSOME'
Select an activity: 2

  ********** Modify entry address book *********
Record No.         17
Select an activity:
                         ✓
Record No.         17
Select and activity: 33          +4

        ****************[mainmenu + update]****************

Enter number for desired selection: 7

        ****************[main-menu]****************
Enter number for desired selection: 6

04/14/86
01:47:00
Enter item number you just finished: 4
                                                  #5
If last item in task, tell monitor
Press any key to continue...K
        ****************[main-menu]****************
```

Bibliography

Allen, Robert B., "Cognitive Factors in Human Interaction with Computers", In A. Badre and Ben Shneiderman (Eds), *Directions in Human/Computer Interaction*, Ablex, 1982, pg 1-18.

Apperley, M. D. and R. Spence, "Hierarachical Dialogue Structures in Interactive Computer Systems", Software - Practice and Experience, Vol 13, 1983, pg 777-790.

Baily, James E. and Sammy W. Peason, "Development of a Tool for Measuring and Analyzing Computer User Satisfaction", Management Science, Vol 29, No 5, May 1983, pg 530-545.

Benbasat, Izak and Yair Wand, "A Structured Approach to Designing Human-Computer Dialogues", International Journal of Man - Machine Studies, Vol 21, No 2, Aug 1984, pg 105-126.

Benbasat, Izak, Albert S. Dexter and Paul Masulis, "An Experimental Study of the Human/Computer Interface", CACM, Vol 24, No 11, Nov 1981, pg 752-762.

Bennett, John, L., "Analysis and Design of the User Interface for DSS", In John L. Bennett (Ed), *Building Decision Support Systems*, Addison - Westley Pub Co., 1983, pg 46-64.

Bennett, John, L., "User-Oriented Graphics Systems for Decision Support in Unstructured Tasks, User-Oriented Design of Interactive Graphics Systems", CACM, 1977, pg 3-11.

Bertino, E., "Design Issues in Interactive User Interfaces", Interfaces in Computing, Vol 3, 1985, pg 37-53.

Black, Jack B. and Marc M. Sebrechts, "Facilitating human - computer communication", Applied Psycholinguistics, Vol 2, No 2, 1981, pg 146-177.

Brown, James W., "Contolling the Complexity of Menu Networks", CACM, Vol 25, No 7, Jul 1982, pg 412-418.

Card, Stuart K., Thomas P. Moran and Allen Newell, *The Psychology of Human - Computer Interaction*, Lawrence Erlbaum Associates, Publishers, Hillsdale, New Jersey, 1983.

Carlson, Eric D., "Developing the User Interface for Decision Support Systems", In John L. Bennett (Ed), *Building Decision Support Systems*, Addison - Westley Pub Co., 1983, pg 65-73.

Carroll, John M. and Caroline Carrithers, "Training Wheels in a User Interface", CACM, Vol 27, No 8, Aug 1984, pg 800-806.

Chapanis, Alphonse, "Computers and the Comman Man", *Information Technology and Psychology*, Praeger, 1982, pg 106-132.

Draper, Stephen W. and Donald A. Norman, "Software Engineering for User Interfaces", IEEE Transactions on Software Engineering, Vol SE-11, No 3, Mar 1985, pg 252-257.

Eason, K. D., "Understanding the naive computer user", The Computer Journal, Vol 19, No 1, Feb 1976, pg 3-7.

Edmonds, E. A., "Adaptive Man-Computer Interfaces", In M. J. Coombs and J. L. Alty (Eds), *Computing Skills and the User Interface*, Academic Press, London and New York, 1981, pg 389-426.

Ehrenreich, S. L.,"Query Languages: Design Recommendations Derived from the Human Factors Literature", Human Factors, Vol 23, No 6, December 1981, pg 709-725.

Emory, William C., *Business Research Methods*, Erwin, 1980.

Foley, James and John Sibert, "How to Design User - Computer Interfaces", Tutorial 1, Proc. CHI'83 Human Factors in Computing Systems (Boston, Dec 1983.

Gains, Brian R. and M. L. G. Shaw, "Dialog Engineering", in M. E. Sime and M. J. Coombs (Eds), Design for Human-Computer Communication, Academic Press, London, 1983, pg 23-53.

Gains, Brian R., "The Technology of Interaction - Dialogue Programming Rules", International Journal of Man-Machine Studies, Vol 14, 1981, pg 133-150.

Gilfoil, Davaid Michael, "Tracking Cognitive Learning and Dialogue Preference in Naive Computer Users: A Longitudinal Study", Dissertation, Stevens Institute of Technology, 1984.

Good, Michael D., John A. Whiteside, Dennis R. Wixon, and Sandra J. Jones, "Building a User-Driven Interface", CACM, Vol 27, No 10, Oct 1984, pg 1032-1043.

Greenberg, Saul and Ian H. Witten, "Adaptive personalized interfaces - a question of viability", Behaviour & Information Technology, Vol 4, No 1, Jan-Mar 1985, pg 31-45.

Hall, P. A. V., "Man - Computer Dialogues for Many Levels of Competence", In James A. Larson (Ed), *End User Facilities in the 1980's*, Computer Society Press, 1982.

Hanson, Stephen Jose, Robert E. Kraut and James M. Farber, "Interface Design and Multivariate Analysis of UNIX Command Use", ACM Transactions on Office Information Systems, Vol 2, No 1, Mar 1984, pg 42-57.

Hauptmann, Alexander G. and Bert F. Green, "A comparison of command, menu-selection and natural-language computer programs", Behaviour & Information Technology, Vol 2, No 2, Apr-Jun 1983, pg 163-178.

Hayes, John R., "Three Problems in Teaching General Skills", in J. Segal, S. Chipman and R. Glaser (Eds), *Thinking and Learning*, Hillsdale, N. J., Lawrence Erlbaum, 1986.

Hayes, Phil, Eugine Ball and Raj Reddy, "Breaking the Man - Machine Communication Barrier", Computer, Vol 14, No 3, Mar 1981, pg 19-30.

Hiltz, Starr Roxanne, *Online Communities*, Ablex Publishing Co.,1984.

Innocent, P. R., "Towards Self - Adaptive Interface Systems", International Journal of Man-Machine Studies, Vol 16, 1982, pg 287-299.

Ives, Blake, Margrethe H. Olson and Jack J. Baroudi, "The Measurement of User Information Satisfaction", CACM, Vol 26, No 10, Oct 1983, pg 785-793.

James, E. B., Computing Reviews, Vol 25, No 4, Apr 1984, pg 181-182.

James, E. B., "The User Interface: How We May Compute", In M. J. Coombs and J. L. Alty (Eds), *Computing Skills and the User Interface*, Academic Press, London and New York, 1981, pg 337-372.

Johnson, Paul, "What Kind of Expert Should a System Be?", The Journal of Medicine and Philosophy, Vol 8, 1983, pg 77-97.

Jones, P. F., "Four Principles of Man-Computer Dialog", IEEE Transactions on Professional Communications, Vol PC-21, No 4, Dec 1978, pg 154-159.

Kerber, Kennith W. "Attitudes towards specific users of the computer. Quantitative, decisionmaking and record - keeping applications", Behaviour & Information Technology, Vol 2, No 2, Apr-Jun 1983, pg 187-209.

Kiger, John I., "The Depth/Bredth Trade-Offs in the Design of Menu-Driven User Interfaces", International Journal of Man-Machine Studies, Vol 20, No 2, Feb 1984, pg 201-214.

Kuo, Bob and Ben Konsynski, "An Architecture for Design of Effective IS Dialogues", working paper, University of Arizona,1985.

Larkin, Jill, John McDermott, Dorothea P. Simon and Herbert A. Simon, "Expert and Novice Performance in Solving Physics Problems", Science, Vol 208, Jan 1980, pg 1335-1342.

Larson, James A., *End User Facilities in the 1980's*, Computer Society Press, 1982, pg 1-3, 439-441.

Liang, T. "A Self-Evolving User Interface Design for Decision Support Systems", Proc. of the 17th Annual Hawaii International Conference on Systems Sciences, Jan 1984, pg 548-557.

Liklider, J. C. R., "Man - Computer Symbiosis", I.R.E. Transactions on Human Factors in Electronics, 1960, pg 4-11.

Lucus, Henry, "Use of Computer Based Graphics in Decision Making", Management Science, Vol 27, No 7, July 1981, pg 757-768.

Magers, Celeste S., "An Experimental Evaluation of On-line Help for Non-Programmers", Proc. CHI'83 Human Factors in Computing Systems (Boston), Dec 1983, pg 277-281.

Maguire, Martin, "An Evaluation of Published Recommendations on the Design of Man - Computer Dialogues", International Journal of Man-Machine Studies. Vol 16, 1982, pg 237-261.

Martin, James, *Design of Man-Computer Dialogues*, Prentice-Hall, Inc., 1973.

Martin, Merle P., "Designing DSS for a Broad Range of User Experience", Dissertation, Texas A&M University, Dec 1984.

Mayer, Richard, *The Promise of Cognitive Psychology*, W. H. Freeman and Company, 1981.

Miller, George A., "The Magical Number Seven; Plus or Minus Two: Some Limits on Our Capacity for Processing Information", The Psychological Review. Vol 63, No 2, Mar 1956, pg 52-60.

Mozeico, Howard, "A Human/Computer Interface to Accomodate User Learning Stages", CACM, Vol 25, No 2, Feb 1982, pg 100-104.

Moran, Thomas P., "An Applied Psychology of the User", Computing Surveys, Vol 13, No 1, Mar 1981, pg 1-12.

Morrison, Perry R. and Grant Noble, "Tolerent software for the recognition of user - defined commands in a data base environment", Behaviour & Information Technology, Vol 3, No 1, Jan-Mar 1984, pg 3-12

Nickerson, Raymond S., "Why Interactive Computer Systems are Sometimes Not Used by People Who Might Benefit From Them", International Journal of Man-Machine Studies, Vol 15, No 5, Nov 1981, pg 469-483.

Nievergelt, J, "Errors in Dialog Design and How to Avoid Them", In J. Nievergelt, G. Coray, J. D. Nicond, and A. C. Shay (Eds), *Document Preparation Systems*, North Holland Pub Co., 1982, pg 265-274.

Norman, Donald A., "Stages and Levels in Human-Machine Interaction", International Journal of Man-Machine Studies, Vol 21, No 4, Oct 1984, pg 365-375.

Norman, Donald A., "Design Principles for Human - Computer Interfaces", Proc CHI'83 Human Factors in Computing Systems (Boston), Dec 1983, pg 1-10.

Norman, Donald A., *Learning and Memory*, W. H. Freeman and Company, San Francisco, 1981.

Paxton, Anne Lee and Edward J. Turner, "The Application of Human Factors to the Needs of the Novice Computer User", International Journal of Man-Machine Studies, Vol 20, No 2, Feb 1984, pg 137-156.Potsnak, Kathleen Marie, "Choice of Computer Interface Modes by Empencally Derived Categories of Users", Dissertation, Johns Hopkins University, 1984.

Robertson, Ivan T., "Human information - processing strategies and styles", Behaviour & Information Technology, Vol 4, No 1, Jan-Mar 1985, pg 19-29.

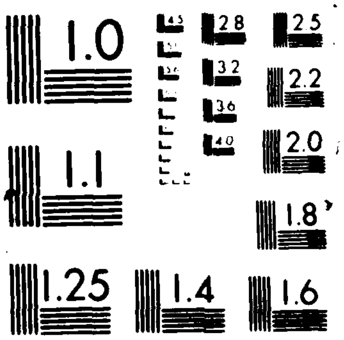
Savage, Ricky E. and James K. Habinek, "A multilevel menu-driven user interface: design and evaluation through simulation", in J. C. Thomas and M. L. Schneider (Eds.), Human Factors in Computer Systems, Ablex Publishing Corp., Norwood, NJ, 1984, pg 165-186.

Shneiderman, Ben, "Human Factors Experiments for Designing Interactive Systems", Computer, Dec 1979, pg 9-19.

Shneiderman, Ben, *Software Psychology: Human Factors in Computers and Information Systems*, Winthrop Publishers, Inc., 1980.

Simon, Herbert A., *The Sciences of the Artificial*, The MIT Press, 1984.

AD-A186 115 USING MULTIPLE DIALOG MODES IN A USER-SYSTEM INTERFACE 3/3
TO ACCOMODATE DIFF..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH R A TAYLOR 1986
UNCLASSIFIED AFIT/CI/NR-87-126D F/G 23/2 NL


1.0
2.8
2.5
3.2
2.2
3.6
4.0
2.0
1.1
1.8
1.25
1.4
1.6

Smith, David C., Charles Icby, Ralph Kimball, and Bill Verplank, "Designing the Star User Interface", Byte, Vol 7, No 4, Apr 1982, pg 242-282

Smith, Hugh T., "Human - Computer Communication", In H. T. Smithland and T. R. G. Green (Eds), *Human Interaction with Computers*, Academic Press, London, 1980, pg 2-38.

Sprague, Ralph H. Jr. and Eric D. Carlson, *Building Effective Decision Support Systems*, Prentice - Hall, Inc., 1982.

Stevens, G. C., "User - friendly computer systems? A critical examination of the concept", Behaviour & Information Technology, Vol 2, No 1, Jan-Mar 1983, pg 3-16.

Tennant, H. R., "Menu - Based Natural Language Understanding", AFIPS Conf. Proc., Vol 53, 1984, pg 631-635.

Thimbleby, H., "Dialogue Determination", International Journal of Man-Machine Studies, Vol 13, 1980, pg 295-304.

Treu, S., "A framework fo characteristics applicable to graphical user-computer interaction", In S. Treu (Ed), User-Oriented Design of Interactive Graphics Systems, ACM/SIGGRAPH, New York, 1977, pg 61-71.

Vassiliou, Yannis, Matthias Jarke, Edward A. Stohr, Jon A. Turner and Norman H. White, "Natural Language for Database Queries: A Laboratory Study", Fourth International Conference on Information Systems, Dec 1983.

Wasserman, A. I., "User Software Engineering and the Design of Interactive Systems", Proceedings of the 5th International Conference on Software Engineering, 1981, pg 387-393.

Winer, B. J., *Statistical Principlals in Experimental Design*, Magraw Hill, 1971, pg 561-563.

Whiteside, John, Sandra Jones, Paula S. Levy and Dennis Wixon, "User Performance with Command, Menu, and Iconic Interfaces", Proc. CHI'85 Human Factors in Computing Systems (San Francisco), Apr 1985, pg 185-191.

Zoltan, E. and Chapanis A., "What do professional persons think about computers?", Behaviour and Information Technology, Vol 1, No 1, Jan - Mar 1982, pg 55-68.

VITA

Roderick Austin Taylor was born in Syracuse , New York, on May 7, 1949, the son of Carolyn Lewis Taylor and Harry Whitter Taylor, Jr. After completing his work at Gen. H. H. Arnold High School, Wiesbaden, Germany, in 1967, he entered Texas A&M University in College Station, Texas. He received the degree of Bachelor of Science in Mathematics from Texas A&M University and was commissioned a 2LT in the United States Air Force in May of 1971. Prior to entering the Air Force, he stayed at Texas A&M University and received a Masters of Computing Science in August of 1972. During the following years, he was employed by the Air Force as a computer systems analyst in various places around the United States and attained the rank of Major. In September, 1982, under Air Force sponsorship, he entered the Graduate School of The University of Texas.

Permanent address : 12602 Terra Nova Lane
Austin, Texas

This dissertation was typed by Roderick A. Taylor.